PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 :
G06F 9/44, 17/30

A1

(11) International Publication Number: WO 99/17193

(43) International Publication Date: 8 April 1999 (08.04.99)

(21) International Application Number: PCT/US98/20419

(22) International Filing Date: 30 September 1998 (30.09.98)

(30) Priority Data:
08/941,438 30 September 1997 (30.09.97) US

(71) Applicant: UNISYS CORPORATION [US/US]; Township Line and Union Meeting Roads, P.O. Box 500, Blue Bell, PA 19424-0001 (US).

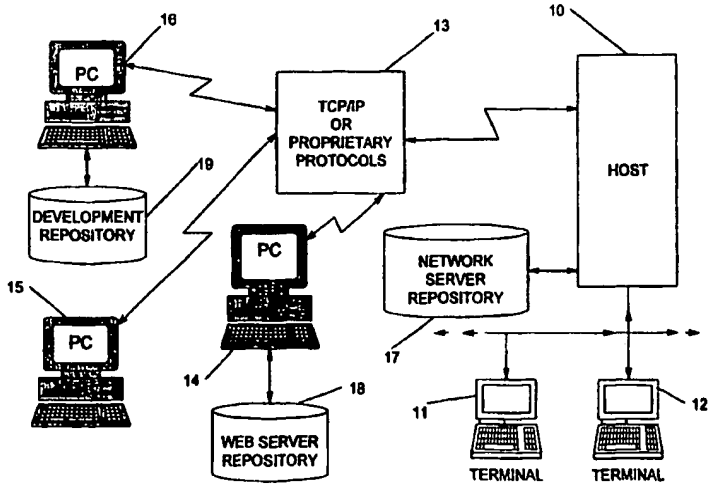
(72) Inventors: MUTSCHLER, Eugene, Otto, III; 39 Maracay, San Clemente, CA 92672 (US). STEFANIAK, Joseph, Peter; 2524, Via Durazno, San Clemente, CA 92673 (US).

(74) Agent: STARR, Mark; Unisys Corporation, Township Line and Union Meeting Roads, P.O. Box 500, Blue Bell, PA 19424-0001 (US).

(81) Designated States: AU, CA, JP, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: A METHOD FOR EXTENDING THE HYPERTEXT MARKUP LANGUAGE (HTML) TO SUPPORT A GRAPHICAL USER INTERFACE CONTROL PRESENTATION



(57) Abstract

A method useful in a computing system that has at least one server (10) and a multiplicity of clients (15, 16) coupled thereto by means of a network (13). The server (10) has a repository (17) coupled to it for storing SCL of a legacy Form and at least one of the clients (15, 16) executes a Web browser program. The method operates in the same client (15, 16) that the Web browser operates and is invoked (151) by the Web browser program for supporting the exact reproduction of the legacy Form. SCL Text is parsed (152) to obtain a sequence of SCL Constructs, some of which contain information relating to GUI Controls of the legacy Form. For each SCL Construct with a GUI Control to be displayed (153), a GUI Control is painted (154) by using type and positioning information derived from each of the SCL Constructs, respectively. Font information from the SCL Constructs is used (155) to set font characteristics of the GUI Control and user inputs are interpreted to navigate from GUI Control to GUI Control. Finally, the user inputs are interpreted (156) to invoke a process that sends contents of the GUI Controls to the host application (157).

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

TITLE: A METHOD FOR EXTENDING THE HYPERTEXT MARKUP LANGUAGE (HTML) TO SUPPORT A GRAPHICAL USER INTERFACE CONTROL PRESENTATION

A portion of the disclosure of this patent document contains material that is subject to copyright protection. The copyright owner has no objection to the facsimile reproduction by anyone of the patent disclosure, as it appears in the Patent and Trademark Office patent files or records, but otherwise reserves all copyright rights whatsoever.

FIELD OF THE INVENTION:

The present invention generally relates to client/server computing in a World Wide Web ("Web") environment, and more particularly to a method for supporting enterprise application binding within a Web Browser.

BACKGROUND OF THE INVENTION:

With the rising popularity of client/server computing and the Web, businesses are looking for even better ways to increase their competitive advantage. Information is one of business's most precious commodities. Accordingly, there is a need for flexibility to position information in ways that best support business organizations and their customers.

Client/server technology offers graphical user interfaces (GUI's) a choice of open systems, rapid application development, increased end-user productivity and much more. By combining this technology with the Internet and intranets, a powerful system is made available to distribute information throughout the business and customer communities. The world Wide web is a purely client/server environment that can bring business to customers. Even if a business has information that needs to be kept within the organization, the Web technology is still available as an Intranet, which is a web site behind a firewall and made available only to employees of the organization. The term "firewall" as used herein refers to a combination of hardware and software that prevents access to secured data over an Internet or Intranet. The idea is to protect a cluster of more loosely administered machines hidden behind the firewall from hackers.

There is also a need to make the move to client/server and Internet technologies without having to

migrate from existing host applications. The best way to build new client/server applications that can be integrated with current applications is to combine the dependability of enterprise-wide systems with the flexibility of distributed processing.

Early attempts at capitalizing on the advantages of the client/server applications involved the use of the PC as a "dumb terminal", or a character-based network terminal, that interacted directly with legacy programs operating on a mainframe or host computer. This approach was unsatisfactory because the user was limited to a character-mode display. Later attempts included such solutions as the Designer Workbench (which is a software tool now referred to as PowerClient and available from Unisys Corporation, assignee of this patent). The PowerClient product gives the user the ability to capture forms descriptions that are on the mainframe, and to convert character-based fields into Windows-based visual elements such as edit fields, buttons, etc. A language called SCL (Screen Control Language) was created for describing the visual elements as well as indicating processing that will be performed on the client PC's.

With reference to various types of legacy programs operating on the host, LINC (i.e., "Logical Information Network Compiler") allows Screen Control Language ("SCL") to be generated directly. SCL is created with the use of the PowerClient product for third generation languages (3GL) such as COBOL or ALGOL legacy programs; and, in the case of MAPPER, which is a

third system and language for legacy programs, there is no way to generate SCL. For MAPPER, the user must use a forms designer tool (such as the Development Studio product) for drawing visual elements to generate the required SCL that matches the MAPPER application.

More recently, it was possible to take SCL definitions of a Form and turn them into a Visual Basic or a PowerBuilder executable program. For an amplification of this method reference is made to a copending patent application entitled A METHOD FOR GENERATING CODE FOR MODIFYING EXISTING EVENT ROUTINES FOR CONTROLS ON A FORM, Serial No. 08/803,137, filed February 19, 1997 by the same inventors hereof and assigned to the same assignee hereof. PowerBuilder is a product of PowerSoft Company (which has recently merged with Sybase, Inc. of Emeryville, CA), and Visual Basic is a product of Microsoft Corporation of Redmond, Washington. However, since the Visual Basic and PowerBuilder client builder products generate executables, there is no way to incorporate the presentation functionality they provide into a Web browser, which is a client-side software module for accessing a Web server. Moreover, there is no obvious way to combine client/server computing on the Web with GUI elements.

SUMMARY OF THE INVENTION:

It is therefore an object of the method of the present invention to provide an evolutionary approach to client/server technology so that a user can retain their existing information technology investment in legacy programs, yet move to client/server computing on the Web. A user can increase productivity and reduce training costs because using the method of the present invention does not require any changes to the legacy programs.

A feature of the method of the present invention is that the generated SCL is dynamically embedded within the HyperText Markup Language (HTML) for each legacy form.

Another feature of the method of the present invention is the capability of being invoked by a Web browser (e.g., an ActiveX control or a Java applet) in the process of displaying the browser's HTML for each legacy form.

An advantage of the present invention is that GUI Controls can be placed at exact pixel coordinates within the browser window thereby exactly duplicating the appearance of the original legacy Form, whereas HTML allows no such control.

Another advantage of the method of the present invention is that it allows font handling of a legacy Form within a browser whether or not the font face, point

size or other attributes are supported by the browser itself.

Still other objects, features and advantages of the present invention will become readily apparent to those skilled in the art from the following detailed description, wherein is shown and described only the preferred embodiment of the invention, simply by way of illustration of the best mode contemplated of carrying out the invention. As will be realized, the invention is capable of other and different embodiments, and its several details are capable of modifications in various obvious respects, all without departing from the invention. Accordingly, the drawings and description are to be regarded as illustrative in nature, and not as restrictive, and what is intended to be protected by Letters Patent is set forth in the appended claims.

BRIEF DESCRIPTION OF THE DRAWINGS:

FIG. 1 is a block diagram of a system that could be used to develop files and to execute the same employing the steps of the method of the present invention.

FIG. 2 is a print of a computer screen display from a 3GL legacy application Form for TRIPS, a report for tracking employee expenses, which is convertible by using the method of the present invention.

FIG. 3 is a print of a computer screen display of the same Form for TRIPS after adding visual controls using the Development Studio.

FIG. 4 is a print of a computer screen display after employment of the method of the present invention.

FIGS. 5A through 5C combined Form a flow chart illustrating the process for development of files for use with the method of the present invention.

FIGS. 6A through 6K (intentionally omitting "6I") illustrate a run-time process that includes the steps performed by the method of the present invention.

FIG. 7 is a flow chart of the process of the present invention.

DETAILED DESCRIPTION OF ONE EMBODIMENT:

The method of the present invention integrates the popular Web browsers with legacy applications in order to provide Form presentation capabilities not supported using HTML, which is today's most popular Web language. This means that one can create a unified GUI front-end "look and feel" for even the most complex applications. The disclosed method is ideal for building and interfacing powerful desktop client applications with 3GL and LINC legacy applications.

One embodiment of the method of the present invention, which is referred to as the SCL Web Control

(or "SWC"), enables the Web browser user to view existing legacy application Forms exactly as designed using the Development Studio. The SWC is an ActiveX compliant control used to communicate with either a Netscape or Microsoft Web browser. ActiveX is a software technology sponsored by Microsoft Corporation. In the case of Netscape, the control hosted by a Netscape plug-in. The SWC interprets the SCL forms description language, and is loaded by the Web browser when referenced by an HTML page. The SWC interprets the SCL embedded within the HTML page and paints the Form as exactly drawn in the Development Studio.

At this juncture of the description several definitions are added to assist in the understanding of the present invention.

Application shall mean a program written by a user which accomplishes some operation.

Command shall mean an instruction to perform an operation. In the Web Agent (defined hereafter) context, the Command operation is to obtain ("Open") a host Form or to send the results of an entry of data into the Form back to the application ("Transmit").

Data Name shall mean the name of a host application datum. That is, is the identification given to a field on a host screen.

Data Value shall mean the value of a host application datum. That is, the value associated with a field on a host screen.

Form shall mean the visual display elements of an application containing fields for entry and display of the application.

GUI Control shall mean a visual element of a GUI display, such as an edit box or a list box. Other examples include buttons, list boxes, check boxes and combo boxes.

Host Reply Definition (HRD) shall mean a file used in the PowerClient 3GL environment that maps characters in the data stream from the host with the fields and their Data Names in the modernized Form.

Partition or Partition Name shall mean a space in a repository, thereby providing a means for categorizing data in the repository. In particular, it shall mean a name given to represent a series of Forms associated with a 3GL/LINC (i.e., Legacy) application.

Repository Object shall mean such things as SCL Text (defined hereafter), list box data, an image, etc.

SCL Construct shall mean a valid production of the SCL language's grammar, i.e., an SCL Text (defined hereafter) statement.

SCL Control shall mean an SCL Construct that describes a particular GUI Control, such as an edit field, list box, etc.

SCL Web Control shall mean an ActiveX based control used in a PowerClient run-time environment.

SCL Text shall mean a collection of SCL Constructs. SCL is a Unisys proprietary language. SCL Text is used to describe GUI Controls to be rendered within a browser.

SCL Files shall mean those files that contain SCL Text, which are referred to as an object in a repository.

Script shall mean a sequence of instructions used to establish connectivity with a host computer and a specific application of the host.

Uniform Resource Locator (URL) shall mean an identifier for locating a resource on the Web.

Web Agent shall mean the PowerClient module that interprets special URL requests passed to it by the Web Server (defined hereafter). It uses information in the special URL to obtain data from the host for a particular Form and returns it to the Web Control via the Web Server and Web browser. It is also a series of both client/server components that permit LINC/3GL applications to be accessible over the Web.

Workstation Driver File (WDF) shall mean a file used in the PowerClient LINC environment that maps characters in the data stream from the host with the fields and their Data Names in the LINC Form.

Referring now to FIG. 1, a client/server system configuration is illustrated wherein a host or mainframe 10 has terminals 11 and 12 coupled thereto and network connection 13 coupling to clients 14, 15 and 16. The network connection 13 may typically comprise a TCP/IP or any other proprietary protocol. The host 10 could be any computing system capable of operating in a client/server environment, such as for example any Unisys or IBM computer. The host 10 is coupled to a network server repository 17, the client 14 (also referred to herein as an NT Web Server) is connected to a client repository 18;

and, the client 16, which in the illustrated embodiment is used for development, is coupled to a development repository 19. The clients 14, 15 and 16 could typically be any currently available PC capable of executing the Windows 95 or NT operating system. However, it is required that the client 14 be able to execute the Windows NT operating system.

The host 10 is capable of executing software programs including LINC and 3GL legacy programs. The client 14 is capable of executing software programs including InfoConnect (transports), PowerClient and Web Agents. The client 15 is capable of executing software programs such as SCL Web Control and Web browsers. The client 16 is capable of executing software programs such as PowerClient and InfoConnect. In addition to those programs alluded to hereinabove, the client 16 is capable of executing many of the more popular and widely-used programs such as PowerBuilder and Visual Basic.

The Web Server repository 18 stores Forms that are distributed to clients using Web browsers. The repository 18 is the database that contains LINC Form objects, image objects, Script files and data files that are downloadable from the host system, as required.

The network server repository 17 holds host application Forms, and it provides version control for Development Studio Forms.

The Development Repository 19 is used as a database for the development process and in particular

stores LINC Form objects, image objects, data files and scripts. Moreover, the development repository 19 contains the objects and scripts necessary to develop and properly display Forms in the Web browser. As one begins modernizing their applications, developers can take advantage of the repository 19 to share tasks. When a new Form is stored in this repository, it is immediately available to other development environment users.

The InfoConnect program includes three parts of a suite that runs with PowerClient, and they are the Unisys MT Emulator, the UTE Emulator and the IBM 3270, all of which are available from DCA, Inc., of Cincinnati, Ohio. PowerClient is a program available from Unisys Corporation of Blue Bell, Pa., assignee of the present invention. PowerClient includes 10 components: Code Generation Assistants (two), 3GL Work Bench, LINC Work Bench, MAPPER, Development Studio, Repositories on the Mainframe (two), CBT and Web Agent.

Referring now to FIG. 2, a print of a screen display of a Form for a 3GL legacy application is illustrated. When a client user calls up the INFOconnect and logs onto the server and specifies TRAVEL, this particular Form entitled TRIPS appears on the screen. It is a typical Form to be completed by an employee of an organization for reimbursement of travel expenses. Each field of this Form is depicted by a space between opposing arrowheads. For example, the field entitled EMPLOYEE SSN is that space 20 between opposing arrowheads 21 and 22, and includes a display screen cursor 23. The next field LAST NAME is that space 24 between opposing arrowheads 25

and 26. The remaining fields of the Form are similarly depicted. There are a total of fifteen (15) fields on this Form.

Preparatory to modifying the TRIPS Form using the method of this invention, the user captures this Form using the combined facilities of the INFOconnect Emulator and the PowerClient Development Studio (PDS). The user then creates Data Names for each field of the Form. SCL syntax will contain each of the Data Names created. For example, for the Form shown in FIG. 2, the following information will be created and stored in the development repository 19 and is to be used in the SCL generation process. The user would indicate the data entry fields and the static text in the Form.

SCL	SCL OFFSETS		
DATANAMES	Row	Column	Length
EMPSSN	3	19	20
LASTNAME	4	19	25
CREATOR	5	19	10

•
•
•

The user/creator has a great deal of flexibility and can create a push button or a window for a specific field, can change names or use previous text, etc. The output is stored in the development repository 19.

Referring now to FIG. 3, a print illustrates a computer screen display of a modernized version of the 3GL

Form shown in FIG. 2 and described hereinabove. Notice that this modernized Form includes windows for entering data, wherein window 20 corresponds to the space 20 in FIG. 2 for entry of an Employee SSN. Window 24 corresponds to the space 24 in FIG. 2. A TRANSMIT button 28 is added by the modernizing process.

Referring now to FIG. 4, the modernized Form shown in FIG. 3 is now embedded within a Web browser display, having been stored in the repository 18 and retrieved by a Web browser at client 15 using the method of the present invention. Corresponding reference numbers identify like components.

Referring now to FIG. 5A, which is the first of a three-sheet drawing showing a flow chart of the process for development of files for use with the method of the present invention. The process begins in the client 16 with a start bubble 50 followed by an inquiry as to whether or not Forms for the host application exist in the development repository 19 (diamond 51). If the answer to this inquiry is yes, then the Development Studio is invoked (block 52). On the other hand, if the Forms do not exist in the development repository, another inquiry is made as to whether or not Forms for the host application exist in the host repository 17 (diamond 53). If the Forms do not exist on the host, then the Development Studio is invoked (block 54) and with the use of the Development Studio, the host application is invoked (block 55).

If the host application Form does exist in the host repository 17 (yes leg of the diamond 53), then the PowerClient software is used to request a download of the Forms and the corresponding HRD/WDF files from the host repository 17 (block 56). Next, the Forms and the HRD/WDF files are placed in the development repository 19 (block 57). After this, the Development Studio is invoked (block 58). Upon completion of the step 52 or the step 58 a branch is made to a later point in the process at a connector A, which will be described shortly. Upon completion of the step 55 a branch is taken to FIG. 5B at a connector B.

Referring now to FIG. 5B at the connector B, the next step in the development process is to capture a Form by interaction with the host application (block 60). For the Form thus captured, an SCL Text and an HRD file are generated using the Development Studio (block 61). Following this, the SCL Text and the HRD file are placed in the development repository 19 (block 62). An inquiry is then made as to whether or not there are more Forms (diamond 63). If the answer is yes, then a return is made back to the process block 60. This cycle repeats itself until all Forms have been processed, and an exit is taken on the no leg of the diamond 63.

The process of updating and modifying of the Forms begins when an inquiry is made as to whether or not there are more Forms in the Partition to be modified (diamond 64). Note that the connector A from the preceding FIG. 5A is also an input to the diamond 64. If the answer to the diamond 64 inquiry is yes, then the

developer or user opens the Form by using the Development Studio (block 65). On the other hand, if there are no more Forms in the Partition to be modified, then a branch is made to a later point in the process as illustrated by a connector C. Next, the developer modifies the GUI Controls as desired (block 66). Following this, the process illustration continues on the next sheet of the drawings, FIG. 5C in particular, as illustrated by a connector D.

Referring now to FIG. 5C at the connector D, an inquiry is made as to whether or not there are more Data Names in the Form (diamond 68). If the answer to this inquiry is yes, then the developer verifies that there is an associated GUI Control and assigns a Data Name to the GUI Control (block 69). Once all of the Data Names in the process have been verified, then an exit is taken from the no leg of the diamond 68 to another step in which the developer uses the Development Studio to generate a new SCL Text (block 70). Following this, the developer uses the Development Studio to place the new SCL Text in the development repository 19 (block 71). On completion of this step, a return is made back to that part of the process illustrated in FIG. 5B at the connector A, and in particular to the inquiry in the diamond 64. Once all of the Forms in the Partition have been modified, then an exit is taken from the no leg of the diamond 64 at the connector C and continues in FIG. 5C, whereupon the Partition containing the SCL Files and the HRD/WDF files is exported from the Development Repository 19 (block 72). The exported Partition is

imported into the Web Server repository 18 (block 73), and the process ends (bubble 74).

At this juncture the SCL Text for the Forms in the last application have been developed in accordance with the directions of the developer/user.

Referring now to FIG. 6A, the first of a ten-sheet drawing is shown that illustrates the run-time process, which includes the steps of the present invention. The process begins in the client 15 with a start bubble 80 followed by a process step performed in response to entry or selection of an URL that contains certain elements, among which are an open Command, a Form name, a Script name and a Partition name. A Web browser forwards the URL to the Web server 14 without otherwise interpreting it (block 81). Next, within the Web Server 14, the URL is passed to the Web Agent (to which it is addressed) and awaits for a response from the Web Agent (block 82).

The Web Agent then parses the URL from the Web Server 14 to obtain the Command which is embedded within it (block 83). Next, an inquiry is made as to whether or not the Command is "Open" (diamond 84). If the answer to this inquiry is no, then a branch is taken to a later point in the process as depicted by a connector R. On the other hand, if the answer to this inquiry is yes, then the Web Agent further parses the URL (block 85) to obtain the following:

1.) the Partition in the repository 18 to be used; and,

2.) the Script for communicating with the host.

The process illustration continues in FIG. 6B at a connector G.

Referring now to FIG. 6B at the connector G, an inquiry is made as to whether or not an open connection to the application on the host 10 exists (diamond 88). If the answer to this inquiry is yes, then the Web Agent uses the Script information to locate an open connection to the application on the host 10 (block 89). On the other hand, if the answer to this inquiry is no, then the Web Agent uses the Script information to open a host connection to the host 10 (block 90). After completion of either the step 89 or 90, the Web Agent requests Form data for the requested Form over the open connection (block 91). After this, and using the Partition information from the URL, the Web Agent retrieves the SCL Text and the WDF or HRD file developed for the application Form from its repository 18 (block 92). The process illustration continues on the next sheet of the drawings as denoted by a connector H.

Referring now to FIG. 6C at the connector H, the Web Agent parses the WDF or HRD file and associates Data Names from it with corresponding Data Values from the returned application Form data into Data Name/Data Value pairs (block 93). Next, the Web Agent generates an HTML page having an object reference to the SCL Web Control, whose parameters include:

1.) The SCL Text itself;

2.) A list of the Data Name/Data Value pairs; and,

3.) Other information necessary for the display of the SCL and interpretation of the Data Name/Data Value pairs (block 94).

The Web Agent then returns the generated HTML page to the Web server (block 95). Following this, the Web server, which has been waiting for the response from the Web Agent, returns the HTML page to the Web browser (block 96). The process illustration continues in the next sheet of the drawings within the PC 15, as denoted by a connector I.

Referring now to FIG. 6D at the connector I, the Web browser parses the HTML page (block 98). Upon encountering the object reference to the SCL Web Control, an inquiry is made as to whether or not the SCL Web Control is present (diamond 99). If the answer is no, then SCL Web Control is downloaded (block 100). It is pointed out that the method of this invention begins with the next process step and continues to block 126, which will be described hereinafter. Once the SCL Web Control is present, the browser invokes the SCL Web Control and passes to it the parameters from the HTML page and other necessary information, including references to the Web browser's window (block 101). Following this, the SCL Web Control parses the SCL Text parameter passed to it to obtain information relating to the GUI Controls to be displayed (block 102).

An inquiry is next made as to whether or not there are more SCL Constructs with a GUI Control to be displayed (diamond 103). If the answer to this inquiry is no, then a branch is made to a later part of the process, which will be described later, as designated by a connector J. On the other hand, if the answer to this inquiry is yes, then the SCL Web Control uses the control type and positioning information from the SCL Construct to paint a GUI Control of the designated type at the designated position in the window whose reference was passed to it as a parameter (block 104). Next, the SCL Web Control uses font information from the SCL Construct to set font characteristics of the GUI Control (block 105). The process illustration continues in the next sheet of the drawings as denoted by a connector K.

Referring now to FIG. 6E at the connector K, the SCL Web Control uses foreground/background color information from the SCL Construct to set color characteristics of the GUI Control (block 106). Following this step, a return is made back to the diamond 103 (via a connector T to FIG. 6D) for processing the next SCL Construct. Once all of the SCL Constructs have been processed, a branch is taken via the connector J to a process step wherein the SCL Web Control parses the SCL Text parameter passed to it to obtain information relating to Data Names for the GUI Controls to be displayed (block 107). Following this, an inquiry is made as to whether or not there are more constructs that have a Data Name (diamond 108). If the answer to this inquiry is yes, then the SCL Web Control locates the

corresponding GUI Control and makes an association between the Data Name and the GUI Control (block 109).

The SCL Web Control then uses information from the SCL Text and a Data Value from the Data Name/Data Value list to set initial state information and contents of the associated GUI Control (block 110). Next a return is made back to the diamond 108 for processing the next construct. Once all of the constructs have been processed, a branch is taken to a process step wherein the SCL Web Control parses the SCL Text parameter passed to it to obtain references to objects stored in the Web Agent repository 18 (block 111). The process illustration continues on the next sheet of the drawings as denoted by a connector L.

Referring now to FIG. 6F at the connector L, an inquiry is made as to whether or not there are more references to a Repository Object (diamond 113). If the answer to this inquiry is no, then a branch is made to a later-described part of the process as denoted by a connector Q. On the other hand, if the answer to the inquiry is yes, then the SCL Web Control constructs a message to the Web Agent for the object containing:

1.) The name of the object from the SCL Text; and,

2.) The Partition name from the original URL, which was derived in step 81, FIG. 6A.

Within the Web server 14, the Web Agent parses the message received to obtain the name of the requested

object and the name of the Partition in which it resides (block 115). Next, the Web Agent uses the name of the object and the Partition in which it resides to obtain the contents of the object from the Web Server repository 18 (block 116). After this, the Web Agent constructs a response message containing the contents of the object from the Web Server repository 18 (block 117). The process illustration continues in the next sheet of the drawings as denoted by a connector M.

Referring now to FIG. 6G at the connector M, the Web Agent returns the response message to the SCL Web Control, which has been waiting for this response (block 120). Within the client 15, the SCL Web Control extracts the requested object contents from the response message received from the Web Agent and associates it with its Data Name (block 121). Next, the SCL Web Control locates the GUI Control that is associated with the Data Name (block 122). The SCL Web Control then places the contents of the requested object into the GUI Control thus located (block 123). At this juncture of the process a return is made back to the diamond 113 in FIG. 6F, as denoted by a connector L, to process the next reference to a Repository Object.

The connector Q from the no leg of the diamond 113 (FIG. 6F) intercepts the process at this point. In response to inputs by a user of the client 15, the Web control interprets user inputs and uses them to navigate from one GUI Control to another (block 124). The process illustration continues on the next sheet of the drawings as denoted by a connector N.

Referring now to FIG. 6H at the connector N, in response to inputs by a user of the client 15 indicating completion of interaction with the displayed GUI Controls, the Web control interprets such inputs and invokes the process that sends the data to the host application (block 126). This process begins at a process step where the SCL Web Control examines each GUI Control that it has previously painted (block 127). Following this, an inquiry is made as to whether or not there are more GUI Controls associated with a Data Name (diamond 128). If the answer to this inquiry is no, then a branch is made to a later point in the process as denoted by a connector O. On the other hand, if the answer to this inquiry is yes, then the SCL Web Control obtains the contents of the GUI Control and places this value and the Data Name into a buffer (block 129). Upon completion of this step a return is made back to the diamond 128 to inquire again if there are more GUI Controls associated with a Data Name. When all GUI Controls have been completed, an exit is taken to the next sheet of the drawings as denoted by the connector O.

Referring now to FIG. 6J at the connector O, and still within the client 15, the SCL Web Control constructs a URL of type POST, addressed to the web server named in the original URL (derived at step 81, FIG. 6A), containing:

1.) The buffer Data Name/Data Value pairs;
2.) The host application Form name;
3.) A Command ("Transmit" in this case);
4.) Partition name; and,

5.) Script name (block 131).

The SCL Web Control then passes the URL to the Web browser (block 132). Next, the Web browser sends the URL to the Web server 14 (block 133). Within the Web server and in response to the URL from the previous step, the web server passes the URL to the PowerClient Web Agent (to which it is addressed, step 82) and waits for a response (block 134). Next, the Web Agent parses the URL to obtain the Command (block 135). After this, an inquiry is made as to whether or not the Command is "Transmit" (diamond 136). If the answer to this inquiry is no then an error message is issued (bubble 137). Note that the connector R, which is from the diamond 84 (FIG. 6A) and an indication that the Command is not "Open", is another input to the diamond 136. The error message is issued since the Command is neither an "Open" nor a "Transmit", which are the only two choices. On the other hand if the answer to this inquiry is yes, then the process continues on to the next sheet of the drawings as depicted by a connector P.

Referring now to FIG. 6K at the connector P, on determining that the Command is "Transmit", the Web Agent further parses the URL to obtain:

1.) The Data Name/Data Value buffer;

2.) The name of the Form with which this data is associated;

3.) Partition name; and,

4.) Script name (block 140).

The Web Agent then uses the name of the host application Form thus obtained to access the HRD or WDF object for the Form from the Web Agent repository 18 (block 141). Next, using the HRD or WDF files as a guide, the Web Agent extracts the fields from the Data Name/Data Value buffers and constructs a buffer for transmittal to the host application at the host 10 (block 142). After this, the Web Agent sends the host data buffer to the host application using the Script name obtained in step 140. Since the Command is "Transmit" the connection will have been established by a previous "Open" Command (block 143). It waits for a response from the host 10, which response includes the name of the next host application Form. Finally, the Web Agent retrieves the SCL Text and the HRD or WDF file developed for the next host application Form from the repository 18. After completion of this step the process repeats itself by returning to the process step 93 (FIG. 6C) as denoted by the connector H.

The method of the present invention is summarized in FIG. 7 and is useful in a computing system having at least one server and a multiplicity of clients coupled thereto by means of a network, all as illustrated in FIG. 1 hereof. The server has a repository coupled thereto for storing SCL of a legacy Form and at least one of the clients is disposed for executing a Web browser program. The method operates in the same client that the Web browser operates and is invoked by the Web browser program for supporting the exact reproduction of the legacy Form. Beginning with a start bubble 150, the Web browser invokes the SCL Web Control and passes parameters

to it from the HTML page of the browser and other necessary information, including references to the browser's window (block 151). Next, SCL Text is parsed to obtain a sequence of SCL Constructs, some of which contain information relating to GUI Controls of the legacy Form (block 152). For each SCL Construct with a GUI Control to be displayed (diamond 153), a GUI Control is painted by using type and positioning information derived from each of the SCL Constructs, respectively (block 154). Following this, font information from the SCL Construct is used to set font characteristics of the GUI Control (block 155). After all of the SCL Constructs have been processed, user inputs are interpreted to navigate from GUI Control to GUI Control (block 156). Finally, the user inputs are interpreted to invoke a process that sends contents of the GUI Controls to the host application (block 157).

Although the invention has been described with reference to a specific embodiment, this description is not meant to be construed in a limiting sense. Various modifications of the disclosed embodiment as well as alternative embodiments of the invention will become apparent to one skilled in the art upon reference to the description of the invention. It is therefore contemplated that the appended claims will cover any such modifications of embodiments that fall within the true scope of the invention.

WHAT IS CLAIMED IS:

1. In a computing system having at least one server (10) and a multiplicity of clients (15, 16) coupled thereto by means of a network (13), said server (10) having a repository (17) coupled thereto that contains description language of a Form to be displayed and at least one of said clients (15, 16) being disposed for executing a Web browser program, a method operating in said at least one of said clients (15, 16) (hereafter said client) and being invoked (151) by said Web browser program for supporting the exact reproduction of said Form, said method comprising the steps of:

a. parsing (152) said description language of said Form to obtain a sequence of Constructs, some of which contain information relating to GUI Controls of said Form;

b. for each of said Constructs with a GUI Control to be displayed (153), painting said GUI Control (154) by using type and positioning information derived from each of said Constructs, respectively;

c. interpreting user inputs to navigate from GUI Control to GUI Control (156); and,

d. interpreting user inputs to invoke a process that sends contents (157) of said GUI Controls to said server (10).

2. The method as in Claim 1 further including the step of using font information from each of said Constructs to set font characteristics of said GUI Control.

3. The method as in Claim 1 wherein said description language for a Form is Screen Control Language.

4. In a computing system having at least one server and a multiplicity of clients coupled thereto by means of a network, said server having a repository coupled thereto that contains SCL of a legacy Form to be displayed and at least one of said clients being disposed for executing a Web browser program, a method operating in said at least one of said clients (hereafter said client) and being invoked by said Web browser program for supporting the exact reproduction of said legacy Form, said method comprising the steps of:

a. parsing SCL Text to obtain a sequence of SCL Constructs, some of which contain information relating to GUI Controls of said legacy Form;

b. for each SCL Construct with a GUI Control to be displayed, painting a GUI Control by using type and positioning information derived from each of said SCL Constructs, respectively;

c. interpreting user inputs to navigate from GUI Control to GUI Control; and,

d. interpreting user inputs to invoke a process that sends contents of said GUI Controls to said server;

whereby said GUI Controls appear in said Form, when displayed, exactly as they would in a windowed graphical environment.

5. The method as in Claim 4 further including the step of using font information from each of said SCL Constructs to set font characteristics of said GUI Control.

6. The method as in Claim 4 wherein step b thereof includes painting a GUI Control of a designated type at a designated position in a window whose reference is passed as a parameter.

7. The method as in Claim 6 further including the step of interpreting user inputs from said SCL Construct to set font characteristics of said GUI Control.

8. The method as in Claim 4 further including the step of dynamically embedding SCL within the HTML for each said legacy Form.

9. The method as in Claim 4 being invoked by a Web browser.

10. The method as in Claim 9 wherein said method is invoked in the process of displaying said Web browser's HTML for each said legacy Form.

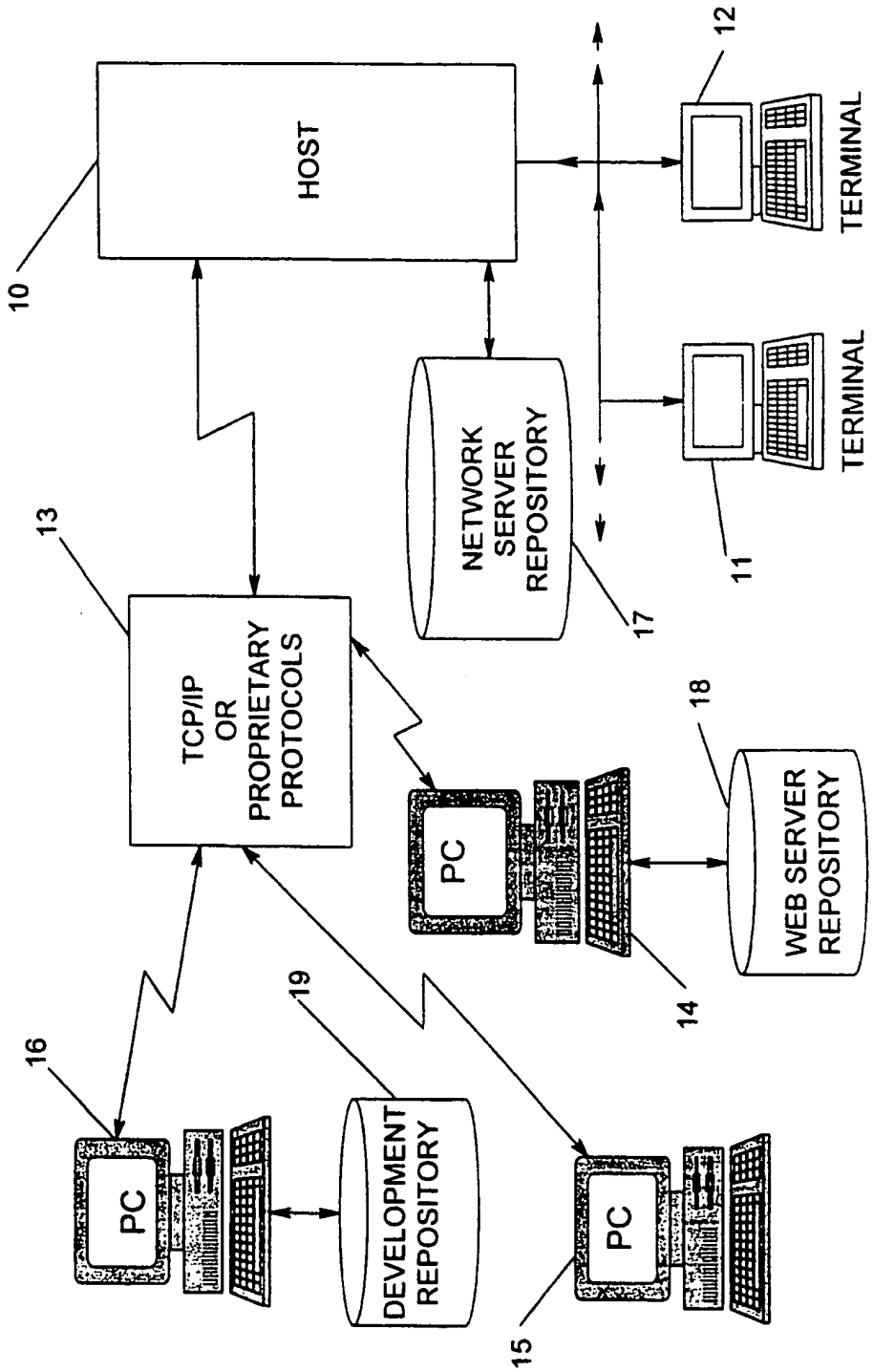

HOST
10
12
TERMINAL
TERMINAL
11
NETWORK SERVER REPOSITORY
17
13
TCP/IP OR PROPRIETARY PROTOCOLS
PC
14
18
WEB SERVER REPOSITORY
16
19
PC
DEVELOPMENT REPOSITORY
PC
15

FIG. 1

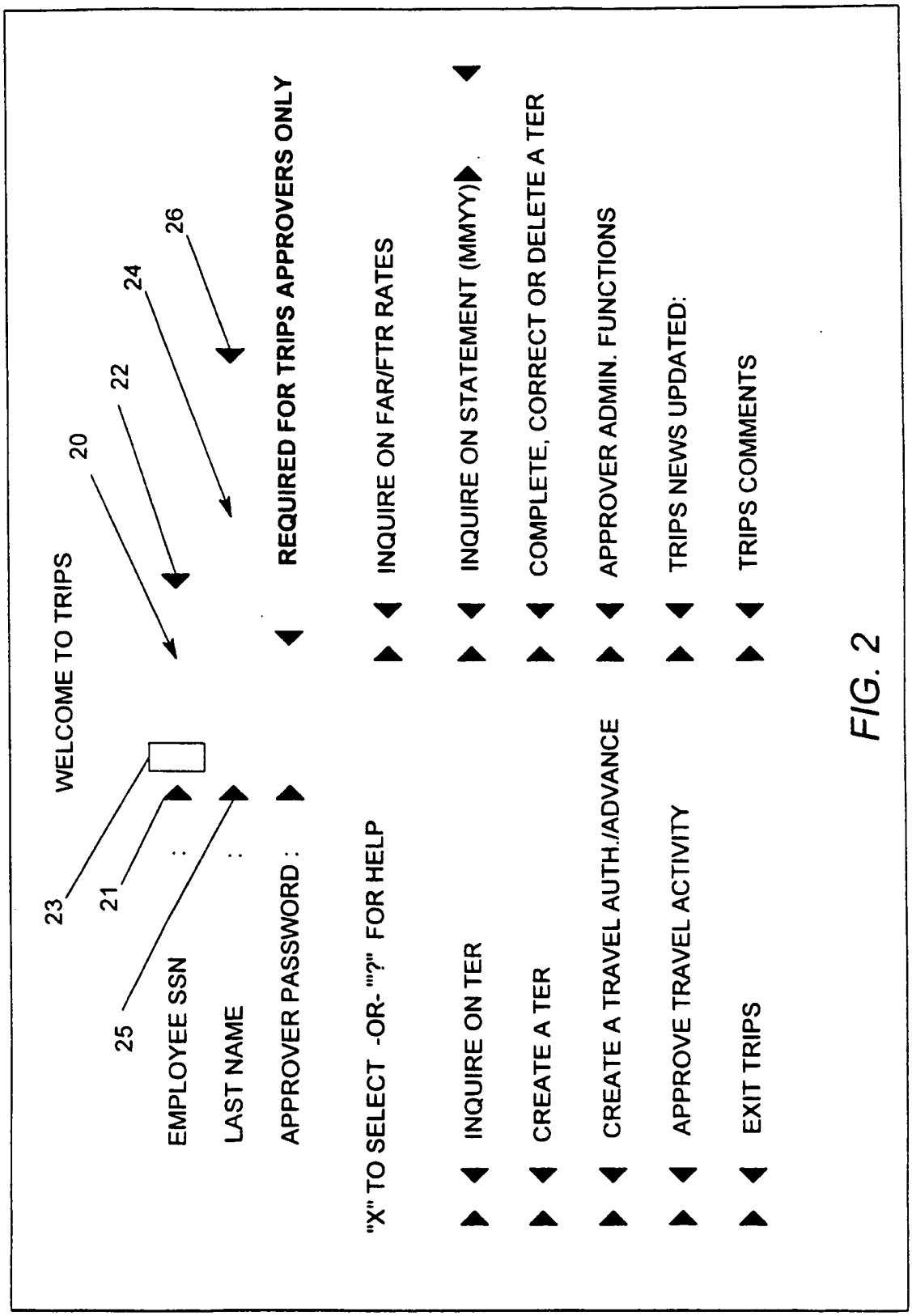

WELCOME TO TRIPS
20
23
21
22
24
26
25
EMPLOYEE SSN : :
LAST NAME : :
APPROVER PASSWORD : ◀ REQUIRED FOR TRIPS APPROVERS ONLY
"X" TO SELECT -OR- "?" FOR HELP ▲ ◀ INQUIRE ON FAR/FTR RATES
▲ ◀ INQUIRE ON TER ▲ ◀ INQUIRE ON STATEMENT (MMYY) ▲ ◀
▲ ◀ CREATE A TER ▲ ◀ COMPLETE, CORRECT OR DELETE A TER
▲ ◀ CREATE A TRAVEL AUTH./ADVANCE ▲ ◀ APPROVER ADMIN. FUNCTIONS
▲ ◀ APPROVE TRAVEL ACTIVITY ▲ ◀ TRIPS NEWS UPDATED:
▲ ◀ EXIT TRIPS ▲ ◀ TRIPS COMMENTS

FIG. 2

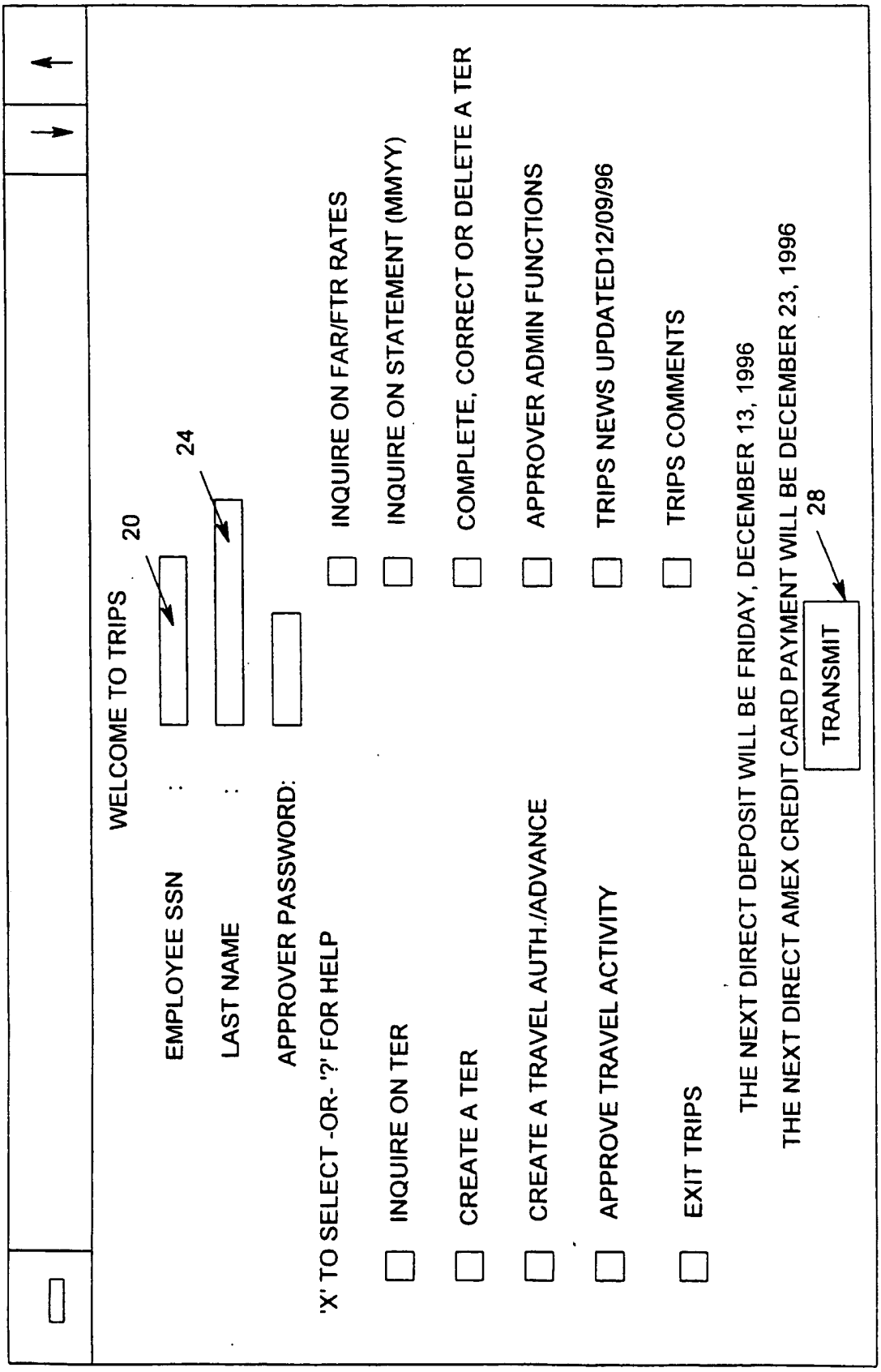

WELCOME TO TRIPS
20
24
EMPLOYEE SSN :
LAST NAME :
APPROVER PASSWORD:
'X' TO SELECT -OR- '?' FOR HELP
INQUIRE ON TER
CREATE A TER
CREATE A TRAVEL AUTH./ADVANCE
APPROVE TRAVEL ACTIVITY
EXIT TRIPS
INQUIRE ON FAR/FTR RATES
INQUIRE ON STATEMENT (MMYY)
COMPLETE, CORRECT OR DELETE A TER
APPROVER ADMIN FUNCTIONS
TRIPS NEWS UPDATED12/09/96
TRIPS COMMENTS
THE NEXT DIRECT DEPOSIT WILL BE FRIDAY, DECEMBER 13, 1996
THE NEXT DIRECT AMEX CREDIT CARD PAYMENT WILL BE DECEMBER 23, 1996
28
TRANSMIT

FIG. 3

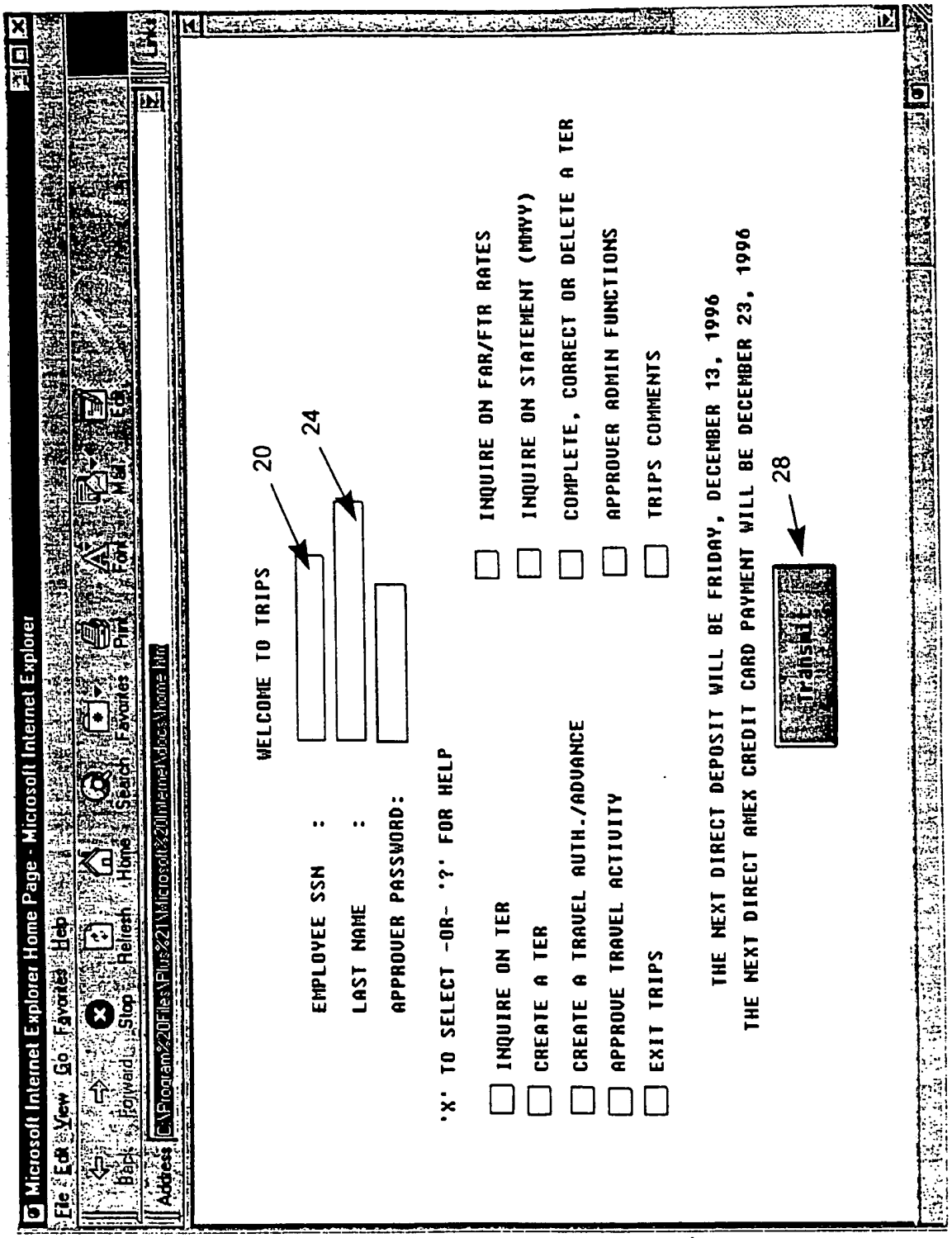

Microsoft Internet Explorer Home Page - Microsoft Internet Explorer
File Edit View Go Favorites Help
Back Forward Stop Refresh Home Search Favorites Print Font Mail Edit
Address C:\Program%20Files\Plus%21\Microsoft%20Internet\Video\Home.htm
WELCOME TO TRIPS
20
24
EMPLOYEE SSN :
LAST NAME :
APPROVER PASSWORD:
'X' TO SELECT -OR- '?' FOR HELP
INQUIRE ON TER
CREATE A TER
CREATE A TRAVEL AUTH./ADVANCE
APPROVE TRAVEL ACTIVITY
EXIT TRIPS
INQUIRE ON FAR/FTR RATES
INQUIRE ON STATEMENT (MMYY)
COMPLETE, CORRECT OR DELETE A TER
APPROVER ADMIN FUNCTIONS
TRIPS COMMENTS
THE NEXT DIRECT DEPOSIT WILL BE FRIDAY, DECEMBER 13, 1996
THE NEXT DIRECT AMEX CREDIT CARD PAYMENT WILL BE DECEMBER 23, 1996
28
Transmit

FIG. 4

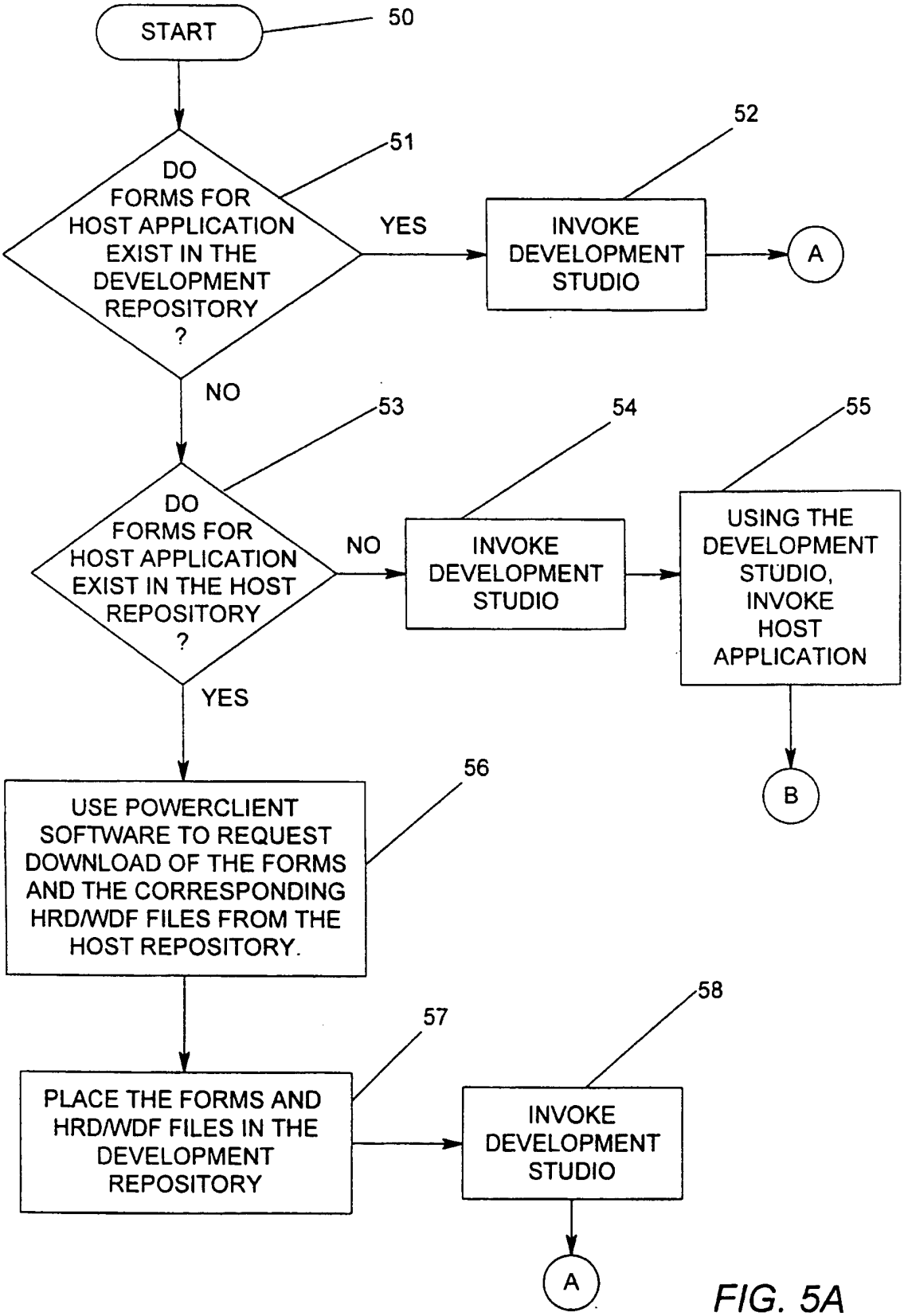

START
50
DO FORMS FOR HOST APPLICATION EXIST IN THE DEVELOPMENT REPOSITORY ?
51
YES
INVOKE DEVELOPMENT STUDIO
52
A
NO
DO FORMS FOR HOST APPLICATION EXIST IN THE HOST REPOSITORY ?
53
NO
INVOKE DEVELOPMENT STUDIO
54
USING THE DEVELOPMENT STUDIO, INVOKE HOST APPLICATION
55
B
YES
USE POWERCLIENT SOFTWARE TO REQUEST DOWNLOAD OF THE FORMS AND THE CORRESPONDING HRD/WDF FILES FROM THE HOST REPOSITORY.
56
PLACE THE FORMS AND HRD/WDF FILES IN THE DEVELOPMENT REPOSITORY
57
INVOKE DEVELOPMENT STUDIO
58
A

FIG. 5A

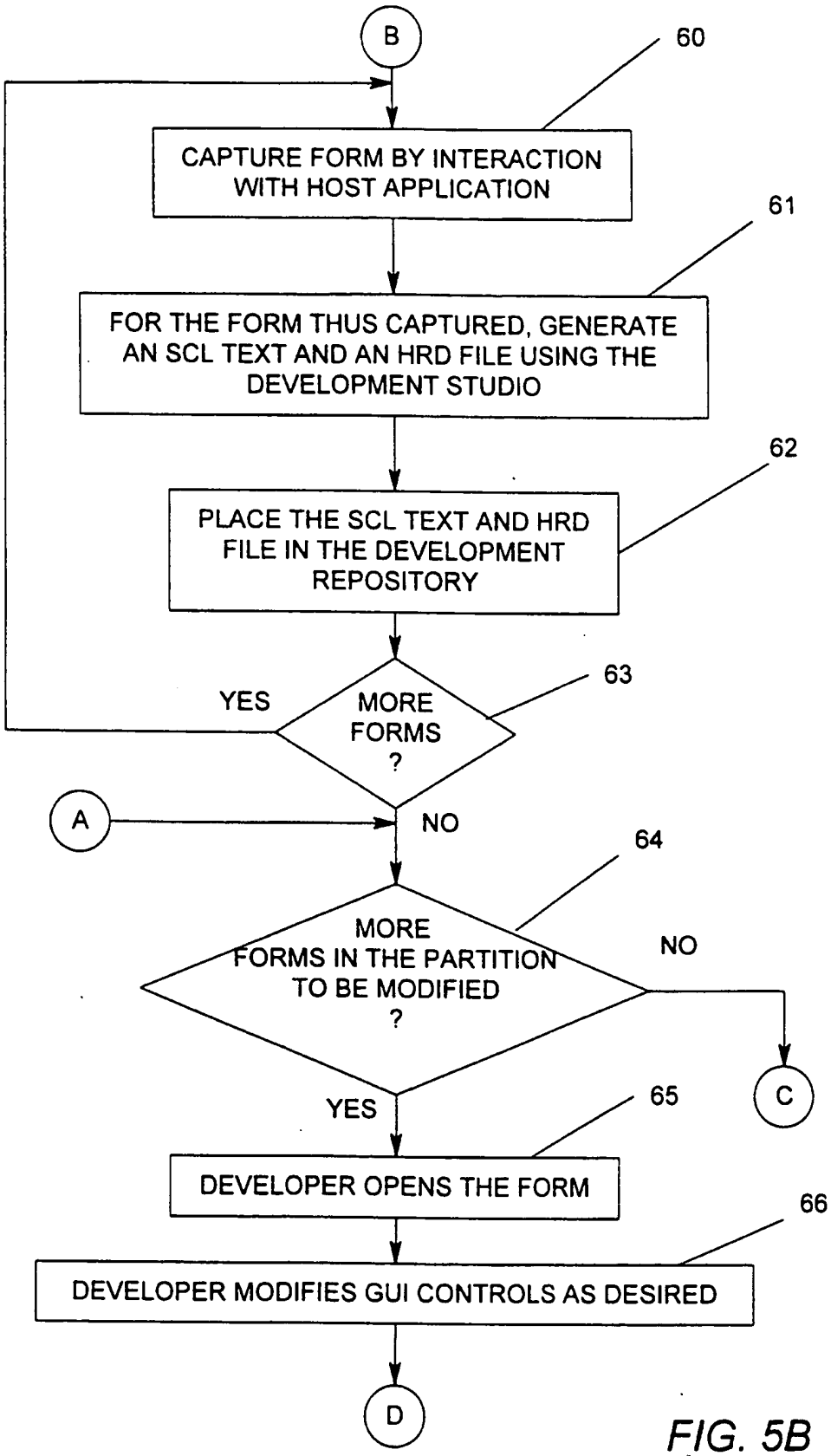

B
60
CAPTURE FORM BY INTERACTION WITH HOST APPLICATION
61
FOR THE FORM THUS CAPTURED, GENERATE AN SCL TEXT AND AN HRD FILE USING THE DEVELOPMENT STUDIO
62
PLACE THE SCL TEXT AND HRD FILE IN THE DEVELOPMENT REPOSITORY
63
MORE FORMS ?
YES
NO
A
64
MORE FORMS IN THE PARTITION TO BE MODIFIED ?
NO
C
YES
65
DEVELOPER OPENS THE FORM
66
DEVELOPER MODIFIES GUI CONTROLS AS DESIRED
D

FIG. 5B

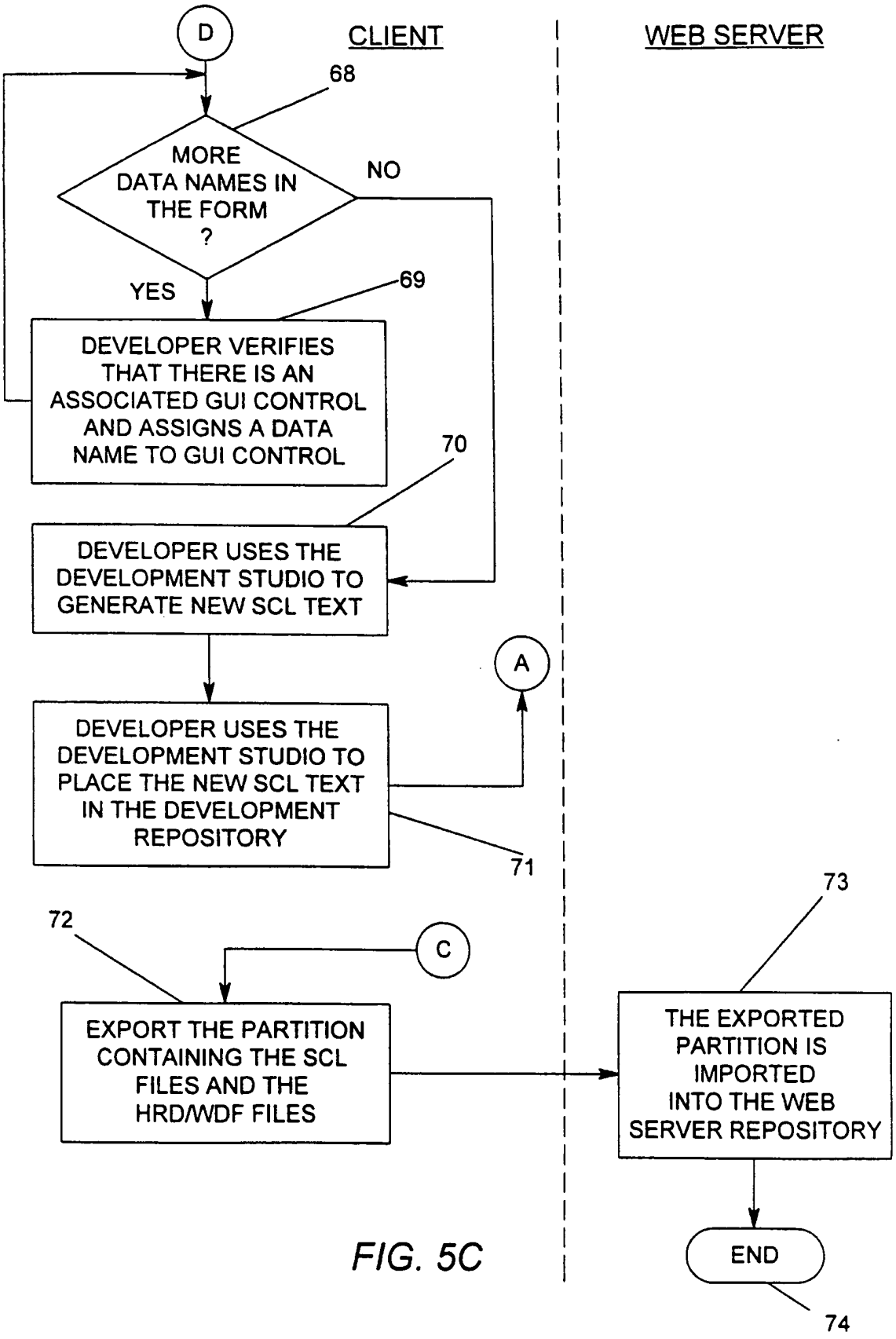

D
CLIENT
WEB SERVER
68
MORE DATA NAMES IN THE FORM ?
NO
YES
69
DEVELOPER VERIFIES THAT THERE IS AN ASSOCIATED GUI CONTROL AND ASSIGNS A DATA NAME TO GUI CONTROL
70
DEVELOPER USES THE DEVELOPMENT STUDIO TO GENERATE NEW SCL TEXT
A
DEVELOPER USES THE DEVELOPMENT STUDIO TO PLACE THE NEW SCL TEXT IN THE DEVELOPMENT REPOSITORY
71
73
72
C
EXPORT THE PARTITION CONTAINING THE SCL FILES AND THE HRD/WDF FILES
THE EXPORTED PARTITION IS IMPORTED INTO THE WEB SERVER REPOSITORY
END
74

FIG. 5C

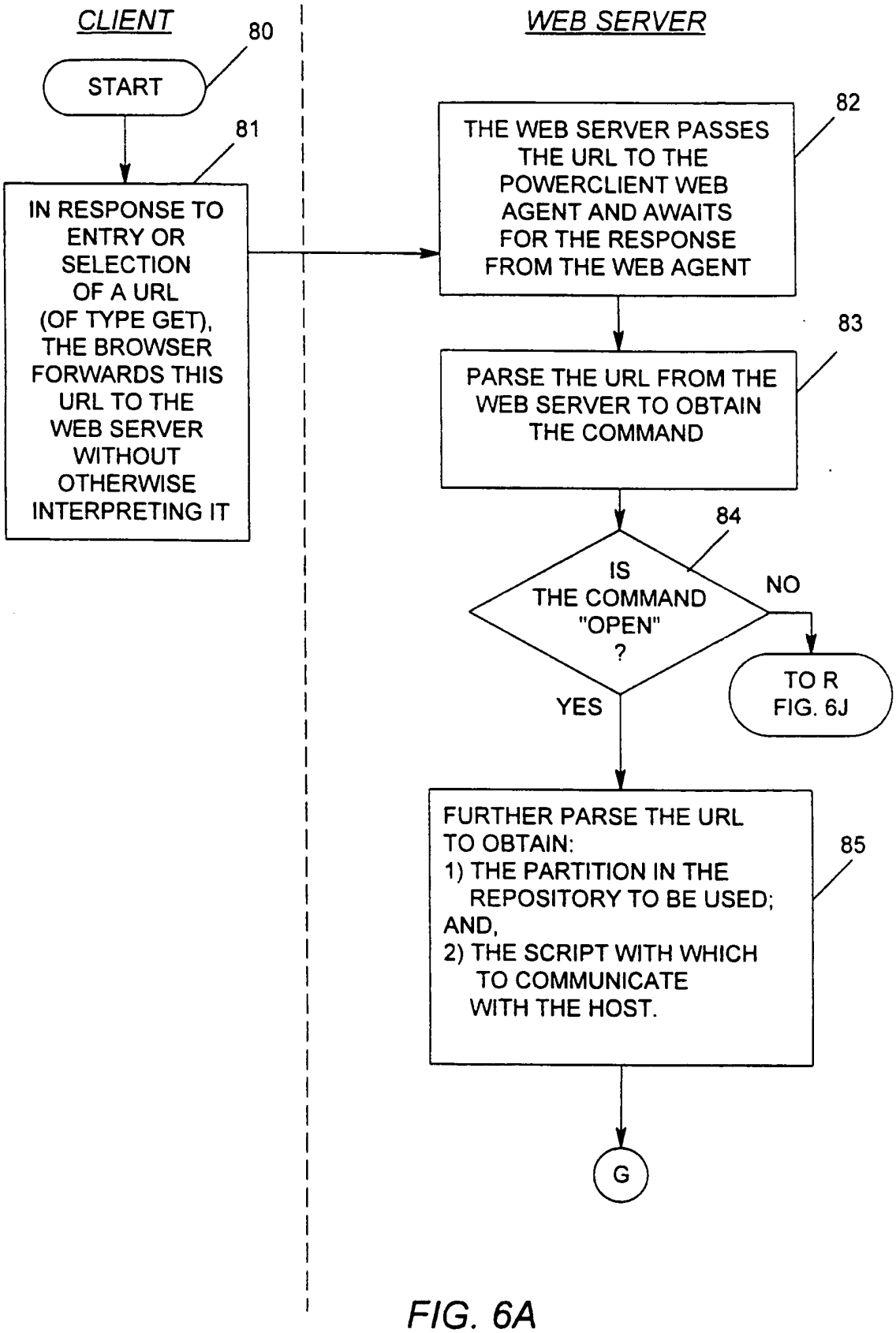

CLIENT
WEB SERVER
80
START
81
IN RESPONSE TO ENTRY OR SELECTION OF A URL (OF TYPE GET), THE BROWSER FORWARDS THIS URL TO THE WEB SERVER WITHOUT OTHERWISE INTERPRETING IT
82
THE WEB SERVER PASSES THE URL TO THE POWERCLIENT WEB AGENT AND AWAITS FOR THE RESPONSE FROM THE WEB AGENT
83
PARSE THE URL FROM THE WEB SERVER TO OBTAIN THE COMMAND
84
IS THE COMMAND "OPEN" ?
NO
TO R FIG. 6J
YES
85
FURTHER PARSE THE URL TO OBTAIN:
1) THE PARTITION IN THE REPOSITORY TO BE USED; AND,
2) THE SCRIPT WITH WHICH TO COMMUNICATE WITH THE HOST.
G

FIG. 6A

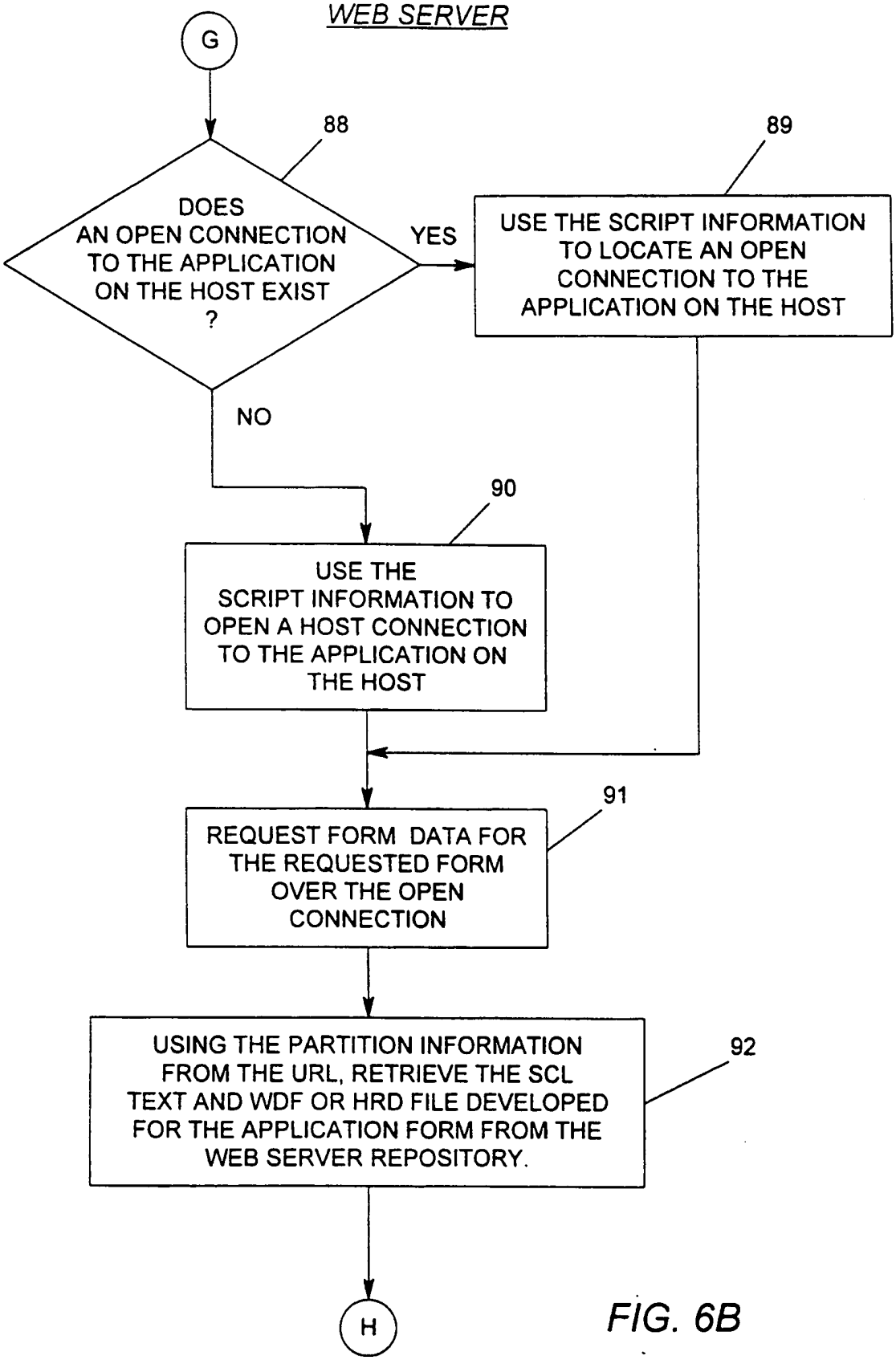

WEB SERVER
G
88
DOES AN OPEN CONNECTION TO THE APPLICATION ON THE HOST EXIST ?
YES
89
USE THE SCRIPT INFORMATION TO LOCATE AN OPEN CONNECTION TO THE APPLICATION ON THE HOST
NO
90
USE THE SCRIPT INFORMATION TO OPEN A HOST CONNECTION TO THE APPLICATION ON THE HOST
91
REQUEST FORM DATA FOR THE REQUESTED FORM OVER THE OPEN CONNECTION
92
USING THE PARTITION INFORMATION FROM THE URL, RETRIEVE THE SCL TEXT AND WDF OR HRD FILE DEVELOPED FOR THE APPLICATION FORM FROM THE WEB SERVER REPOSITORY.
H

FIG. 6B

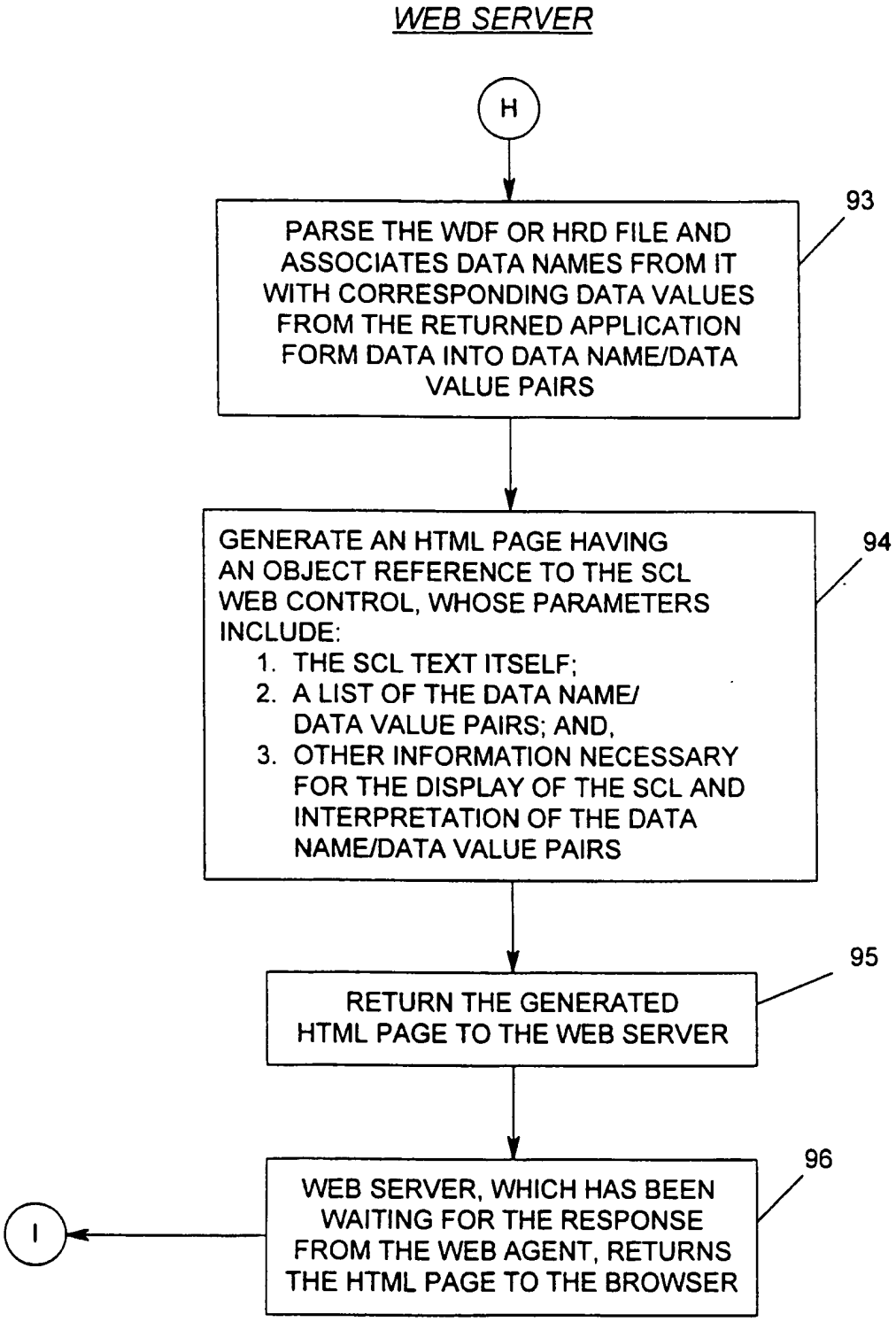

WEB SERVER
H
PARSE THE WDF OR HRD FILE AND ASSOCIATES DATA NAMES FROM IT WITH CORRESPONDING DATA VALUES FROM THE RETURNED APPLICATION FORM DATA INTO DATA NAME/DATA VALUE PAIRS
93
GENERATE AN HTML PAGE HAVING AN OBJECT REFERENCE TO THE SCL WEB CONTROL, WHOSE PARAMETERS INCLUDE:
1. THE SCL TEXT ITSELF;
2. A LIST OF THE DATA NAME/ DATA VALUE PAIRS; AND,
3. OTHER INFORMATION NECESSARY FOR THE DISPLAY OF THE SCL AND INTERPRETATION OF THE DATA NAME/DATA VALUE PAIRS
94
RETURN THE GENERATED HTML PAGE TO THE WEB SERVER
95
WEB SERVER, WHICH HAS BEEN WAITING FOR THE RESPONSE FROM THE WEB AGENT, RETURNS THE HTML PAGE TO THE BROWSER
96
I

FIG. 6C

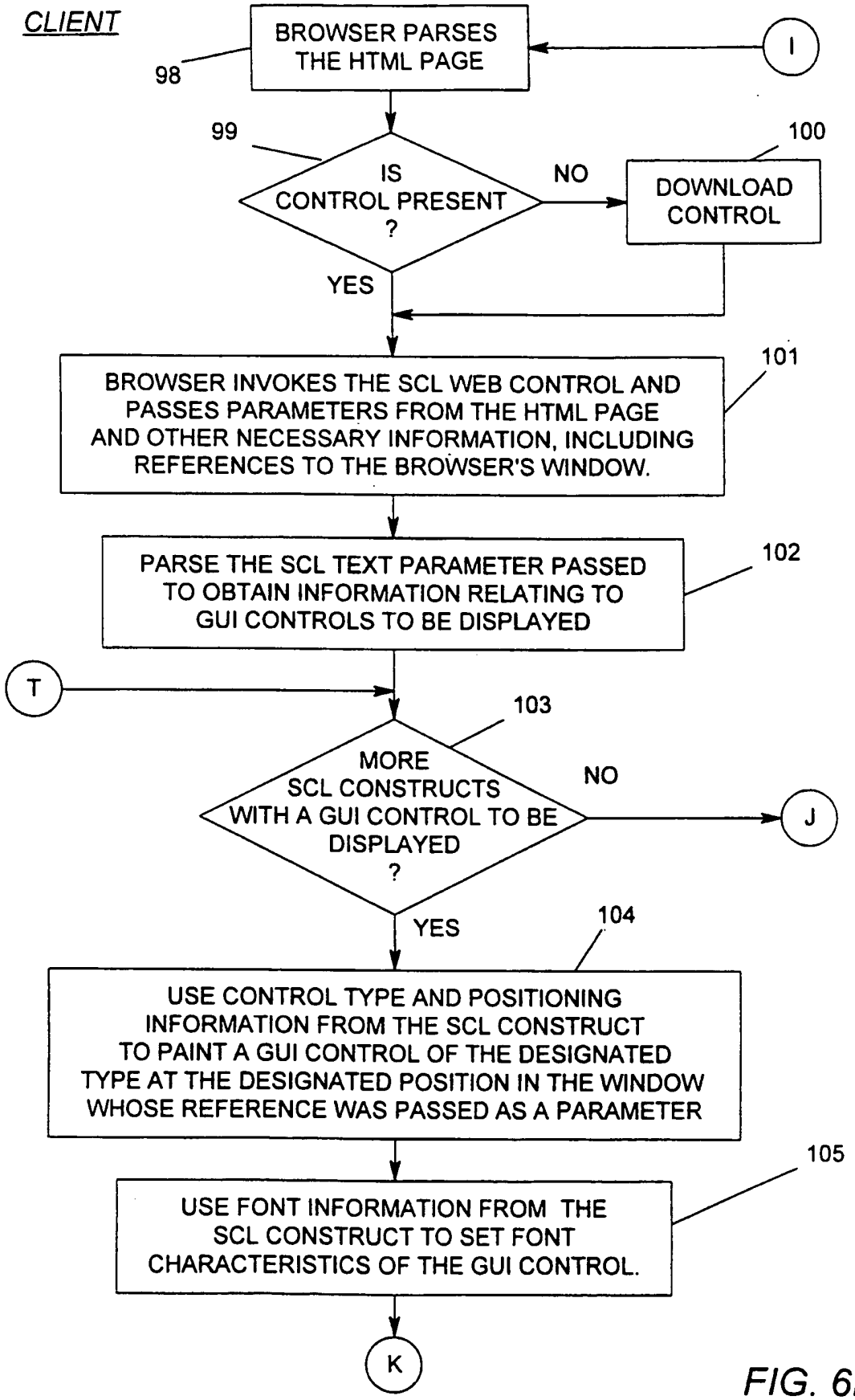

CLIENT
BROWSER PARSES THE HTML PAGE
98
I
99
IS CONTROL PRESENT ?
NO
100
DOWNLOAD CONTROL
YES
BROWSER INVOKES THE SCL WEB CONTROL AND PASSES PARAMETERS FROM THE HTML PAGE AND OTHER NECESSARY INFORMATION, INCLUDING REFERENCES TO THE BROWSER'S WINDOW.
101
PARSE THE SCL TEXT PARAMETER PASSED TO OBTAIN INFORMATION RELATING TO GUI CONTROLS TO BE DISPLAYED
102
T
103
MORE SCL CONSTRUCTS WITH A GUI CONTROL TO BE DISPLAYED ?
NO
J
YES
104
USE CONTROL TYPE AND POSITIONING INFORMATION FROM THE SCL CONSTRUCT TO PAINT A GUI CONTROL OF THE DESIGNATED TYPE AT THE DESIGNATED POSITION IN THE WINDOW WHOSE REFERENCE WAS PASSED AS A PARAMETER
105
USE FONT INFORMATION FROM THE SCL CONSTRUCT TO SET FONT CHARACTERISTICS OF THE GUI CONTROL.
K

FIG. 6D

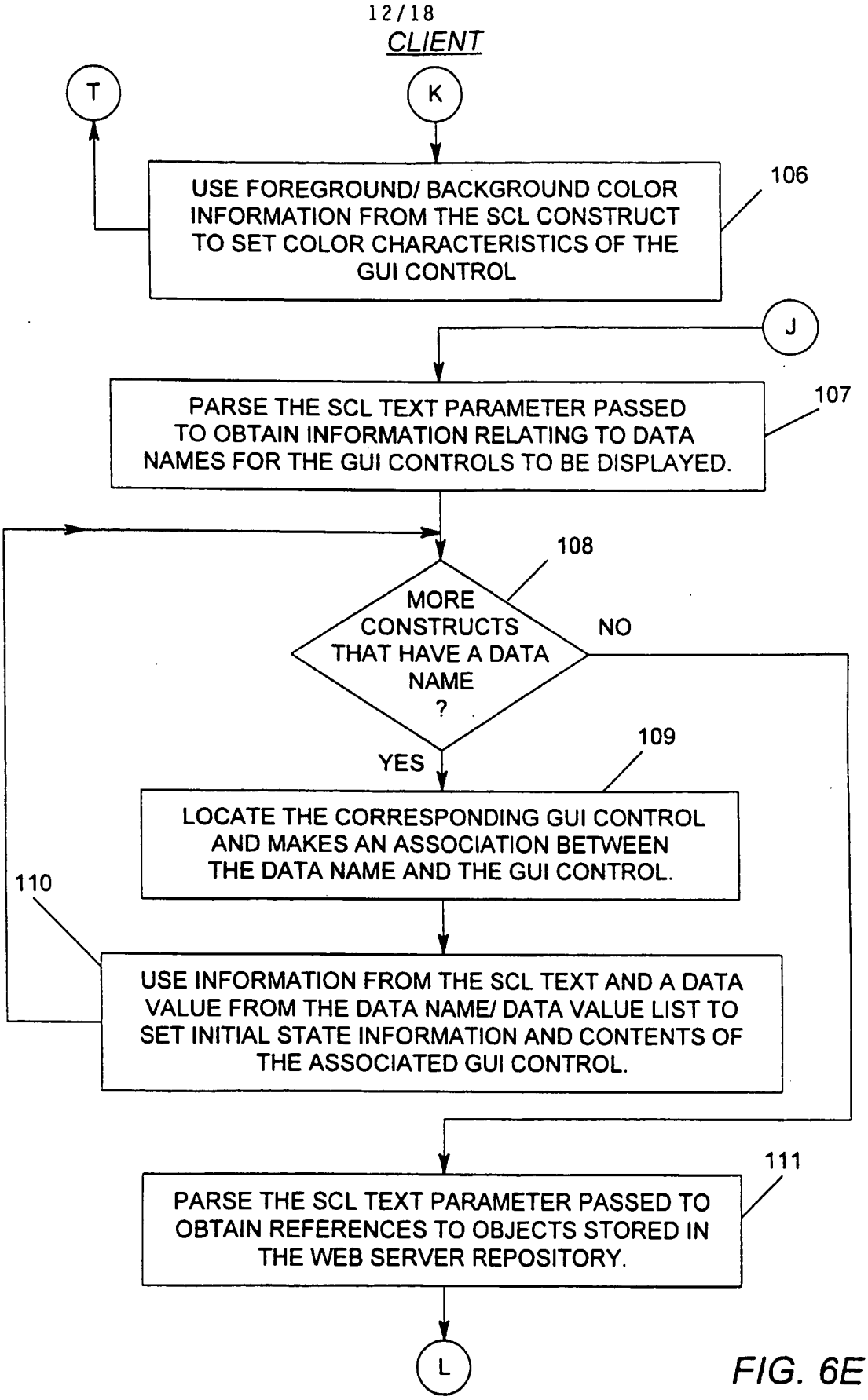

CLIENT
T
K
USE FOREGROUND/ BACKGROUND COLOR INFORMATION FROM THE SCL CONSTRUCT TO SET COLOR CHARACTERISTICS OF THE GUI CONTROL
106
J
PARSE THE SCL TEXT PARAMETER PASSED TO OBTAIN INFORMATION RELATING TO DATA NAMES FOR THE GUI CONTROLS TO BE DISPLAYED.
107
108
MORE CONSTRUCTS THAT HAVE A DATA NAME ?
NO
YES
109
LOCATE THE CORRESPONDING GUI CONTROL AND MAKES AN ASSOCIATION BETWEEN THE DATA NAME AND THE GUI CONTROL.
110
USE INFORMATION FROM THE SCL TEXT AND A DATA VALUE FROM THE DATA NAME/ DATA VALUE LIST TO SET INITIAL STATE INFORMATION AND CONTENTS OF THE ASSOCIATED GUI CONTROL.
111
PARSE THE SCL TEXT PARAMETER PASSED TO OBTAIN REFERENCES TO OBJECTS STORED IN THE WEB SERVER REPOSITORY.
L

FIG. 6E

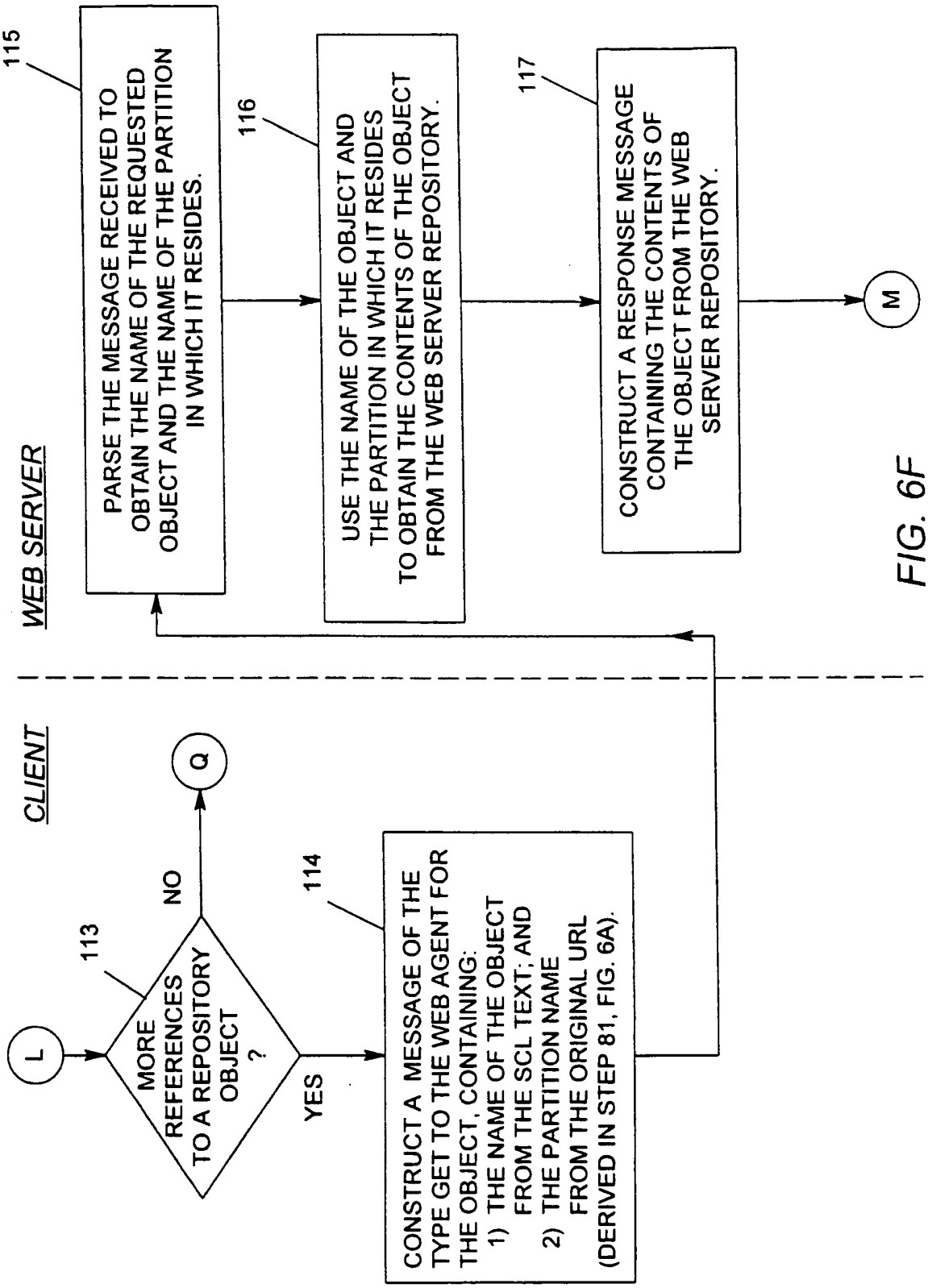

WEB SERVER
CLIENT
L
113
MORE REFERENCES TO A REPOSITORY OBJECT ?
NO
Q
YES
114
CONSTRUCT A MESSAGE OF THE TYPE GET TO THE WEB AGENT FOR THE OBJECT, CONTAINING:
1) THE NAME OF THE OBJECT FROM THE SCL TEXT; AND
2) THE PARTITION NAME FROM THE ORIGINAL URL
(DERIVED IN STEP 81, FIG. 6A).
115
PARSE THE MESSAGE RECEIVED TO OBTAIN THE NAME OF THE REQUESTED OBJECT AND THE NAME OF THE PARTITION IN WHICH IT RESIDES.
116
USE THE NAME OF THE OBJECT AND THE PARTITION IN WHICH IT RESIDES TO OBTAIN THE CONTENTS OF THE OBJECT FROM THE WEB SERVER REPOSITORY.
117
CONSTRUCT A RESPONSE MESSAGE CONTAINING THE CONTENTS OF THE OBJECT FROM THE WEB SERVER REPOSITORY.
M

FIG. 6F

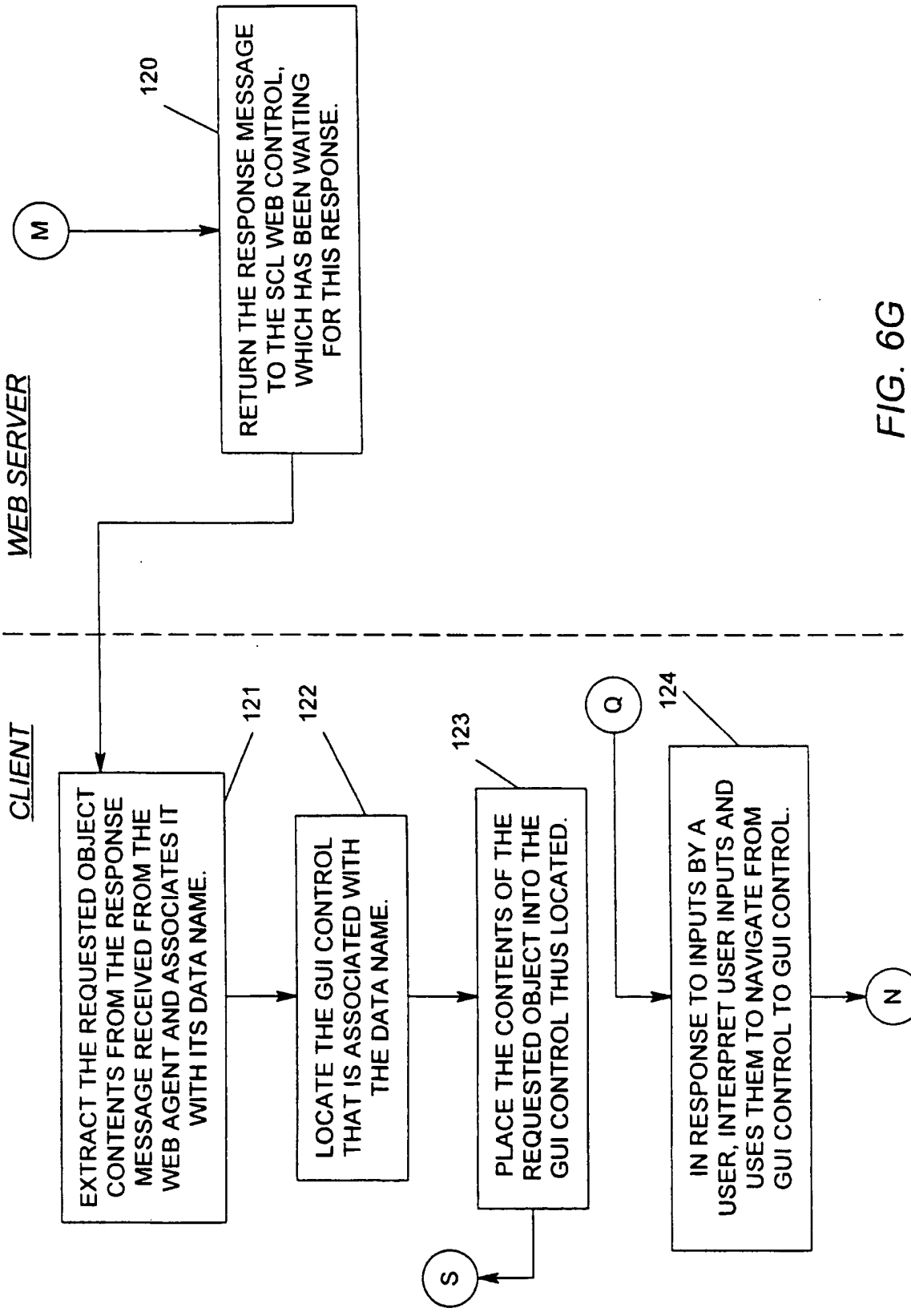

WEB SERVER
CLIENT
M
120
RETURN THE RESPONSE MESSAGE TO THE SCL WEB CONTROL, WHICH HAS BEEN WAITING FOR THIS RESPONSE.
121
EXTRACT THE REQUESTED OBJECT CONTENTS FROM THE RESPONSE MESSAGE RECEIVED FROM THE WEB AGENT AND ASSOCIATES IT WITH ITS DATA NAME.
122
LOCATE THE GUI CONTROL THAT IS ASSOCIATED WITH THE DATA NAME.
123
PLACE THE CONTENTS OF THE REQUESTED OBJECT INTO THE GUI CONTROL THUS LOCATED.
S
Q
124
IN RESPONSE TO INPUTS BY A USER, INTERPRET USER INPUTS AND USES THEM TO NAVIGATE FROM GUI CONTROL TO GUI CONTROL.
N

FIG. 6G

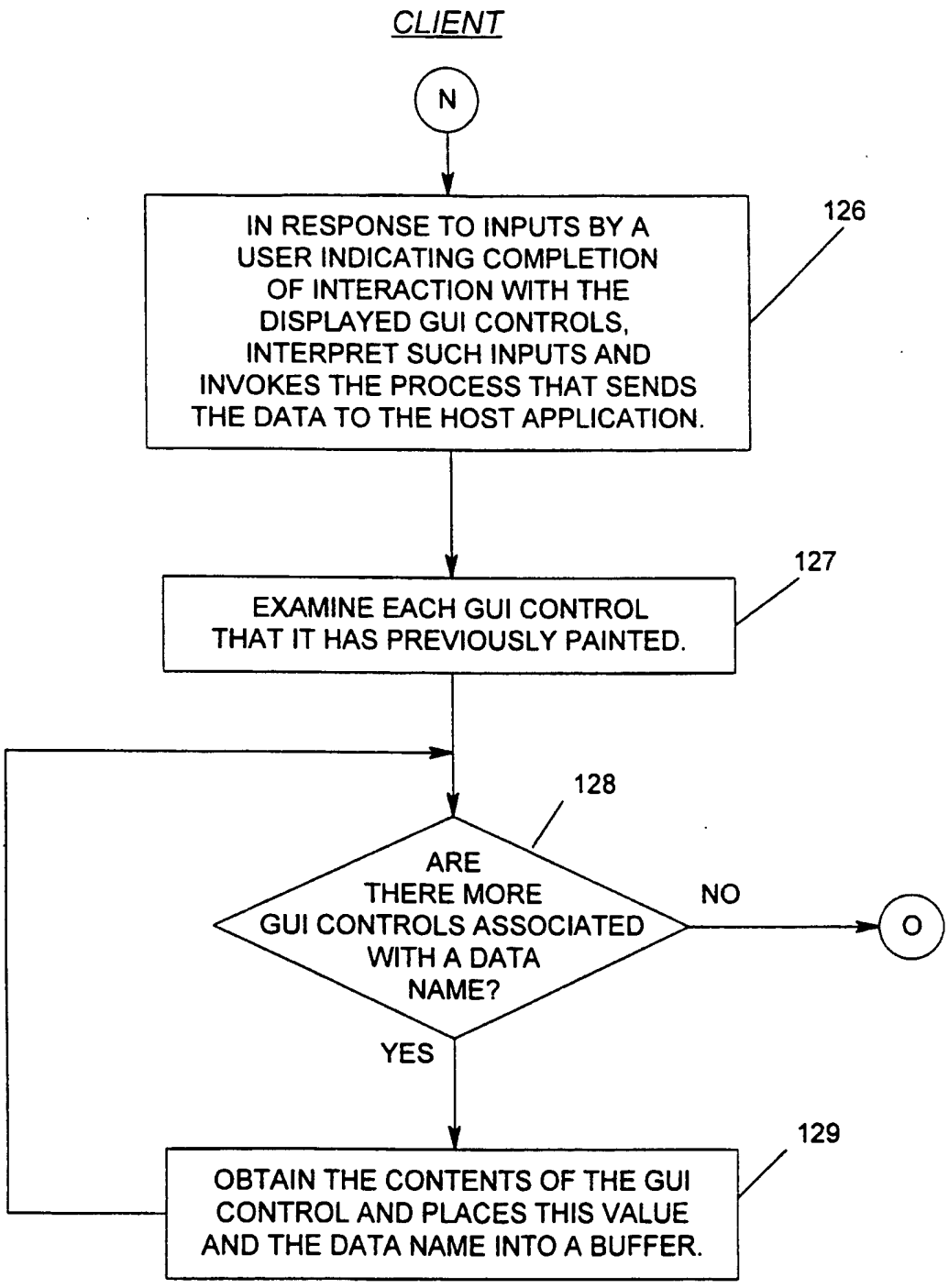

CLIENT
N
IN RESPONSE TO INPUTS BY A USER INDICATING COMPLETION OF INTERACTION WITH THE DISPLAYED GUI CONTROLS, INTERPRET SUCH INPUTS AND INVOKES THE PROCESS THAT SENDS THE DATA TO THE HOST APPLICATION.
126
EXAMINE EACH GUI CONTROL THAT IT HAS PREVIOUSLY PAINTED.
127
128
ARE THERE MORE GUI CONTROLS ASSOCIATED WITH A DATA NAME?
NO
O
YES
OBTAIN THE CONTENTS OF THE GUI CONTROL AND PLACES THIS VALUE AND THE DATA NAME INTO A BUFFER.
129

FIG. 6H

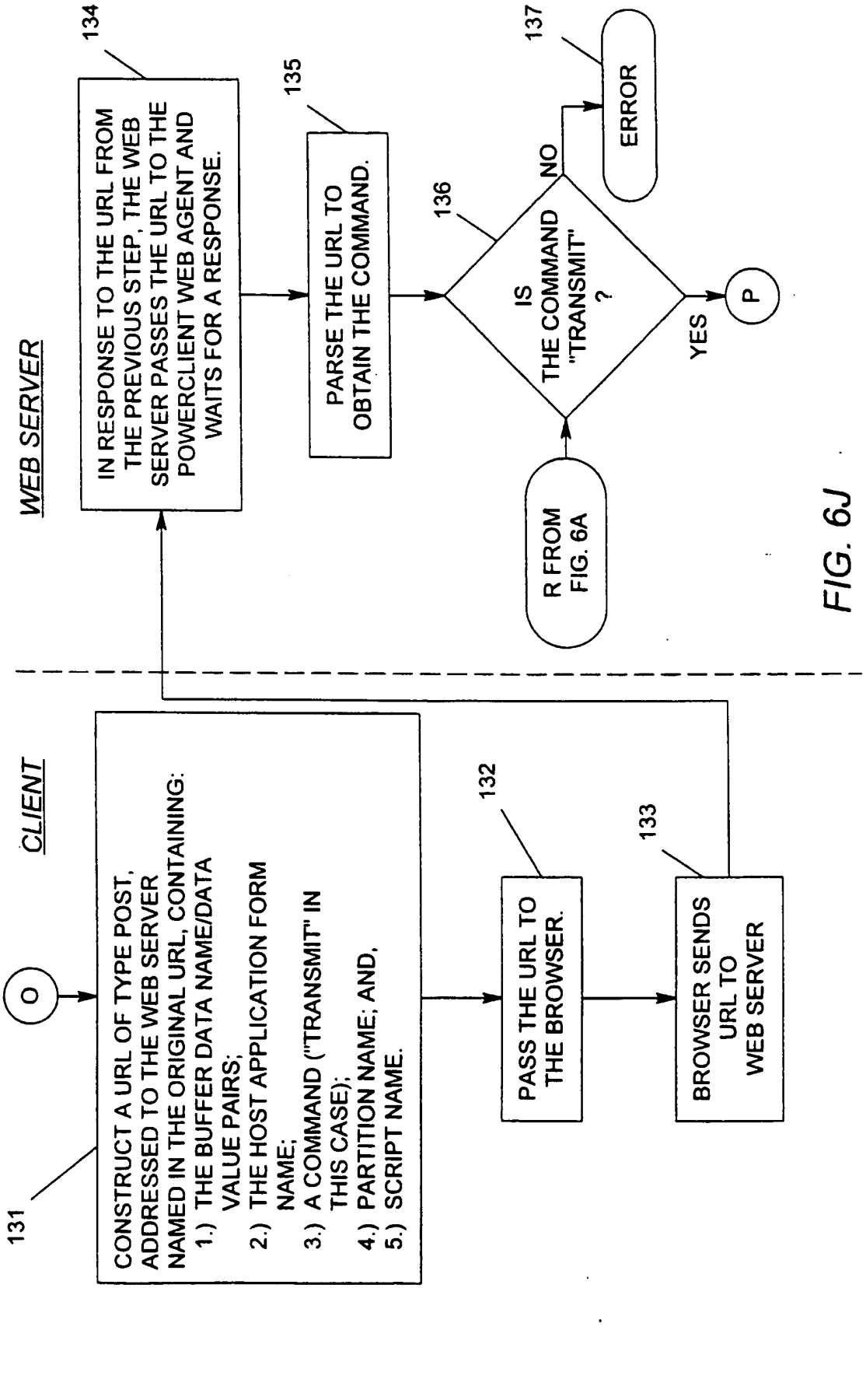

CLIENT
WEB SERVER
O
131
CONSTRUCT A URL OF TYPE POST, ADDRESSED TO THE WEB SERVER NAMED IN THE ORIGINAL URL, CONTAINING:
1.) THE BUFFER DATA NAME/DATA VALUE PAIRS;
2.) THE HOST APPLICATION FORM NAME;
3.) A COMMAND ("TRANSMIT" IN THIS CASE);
4.) PARTITION NAME; AND,
5.) SCRIPT NAME.
132
PASS THE URL TO THE BROWSER.
133
BROWSER SENDS URL TO WEB SERVER
134
IN RESPONSE TO THE URL FROM THE PREVIOUS STEP, THE WEB SERVER PASSES THE URL TO THE POWERCLIENT WEB AGENT AND WAITS FOR A RESPONSE.
135
PARSE THE URL TO OBTAIN THE COMMAND.
136
IS THE COMMAND "TRANSMIT" ?
NO
137
ERROR
YES
P
R FROM FIG. 6A

FIG. 6J

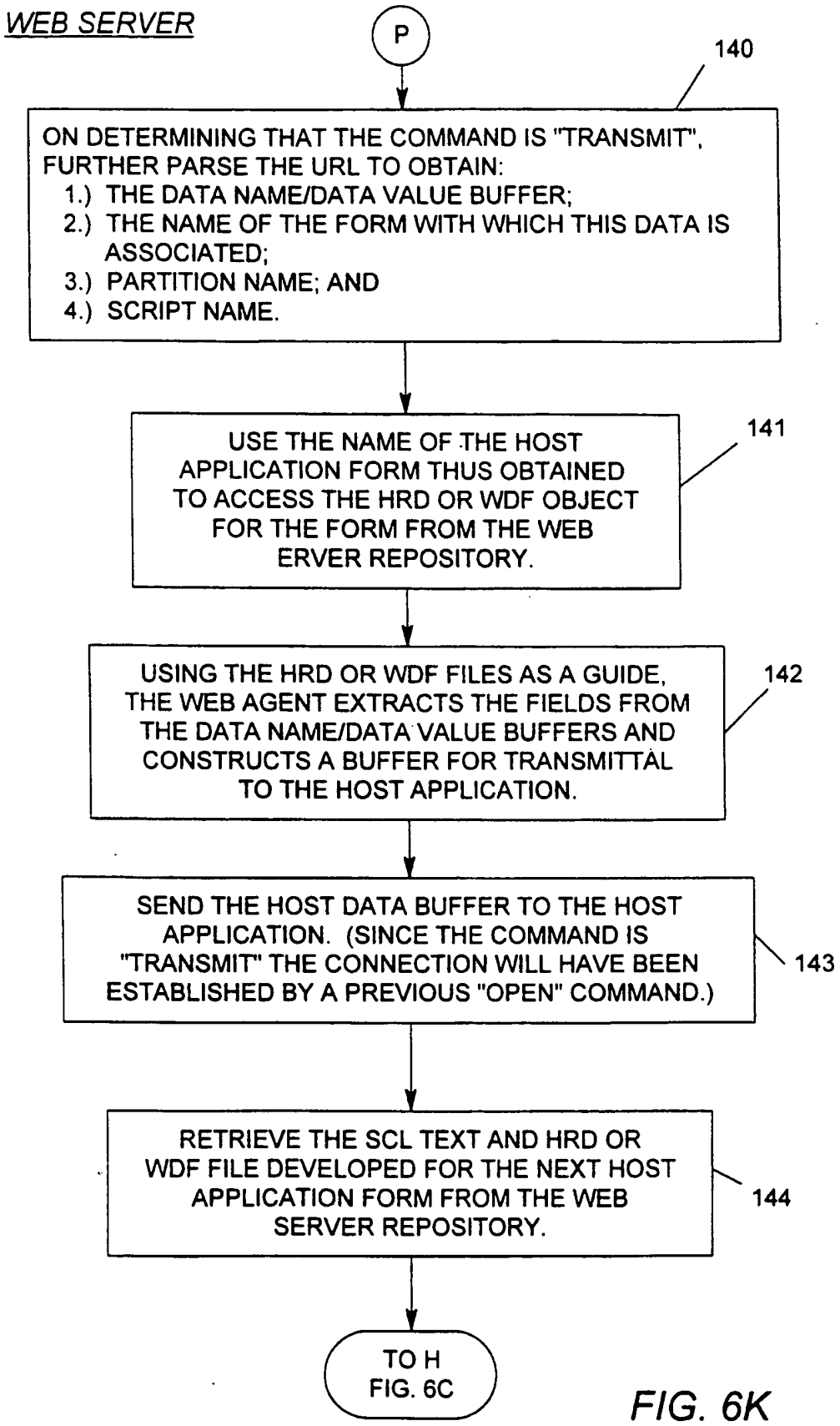

WEB SERVER
P
140
ON DETERMINING THAT THE COMMAND IS "TRANSMIT", FURTHER PARSE THE URL TO OBTAIN:
1.) THE DATA NAME/DATA VALUE BUFFER;
2.) THE NAME OF THE FORM WITH WHICH THIS DATA IS ASSOCIATED;
3.) PARTITION NAME; AND
4.) SCRIPT NAME.
141
USE THE NAME OF THE HOST APPLICATION FORM THUS OBTAINED TO ACCESS THE HRD OR WDF OBJECT FOR THE FORM FROM THE WEB ERVER REPOSITORY.
142
USING THE HRD OR WDF FILES AS A GUIDE, THE WEB AGENT EXTRACTS THE FIELDS FROM THE DATA NAME/DATA VALUE BUFFERS AND CONSTRUCTS A BUFFER FOR TRANSMITTAL TO THE HOST APPLICATION.
143
SEND THE HOST DATA BUFFER TO THE HOST APPLICATION. (SINCE THE COMMAND IS "TRANSMIT" THE CONNECTION WILL HAVE BEEN ESTABLISHED BY A PREVIOUS "OPEN" COMMAND.)
144
RETRIEVE THE SCL TEXT AND HRD OR WDF FILE DEVELOPED FOR THE NEXT HOST APPLICATION FORM FROM THE WEB SERVER REPOSITORY.
TO H
FIG. 6C

FIG. 6K

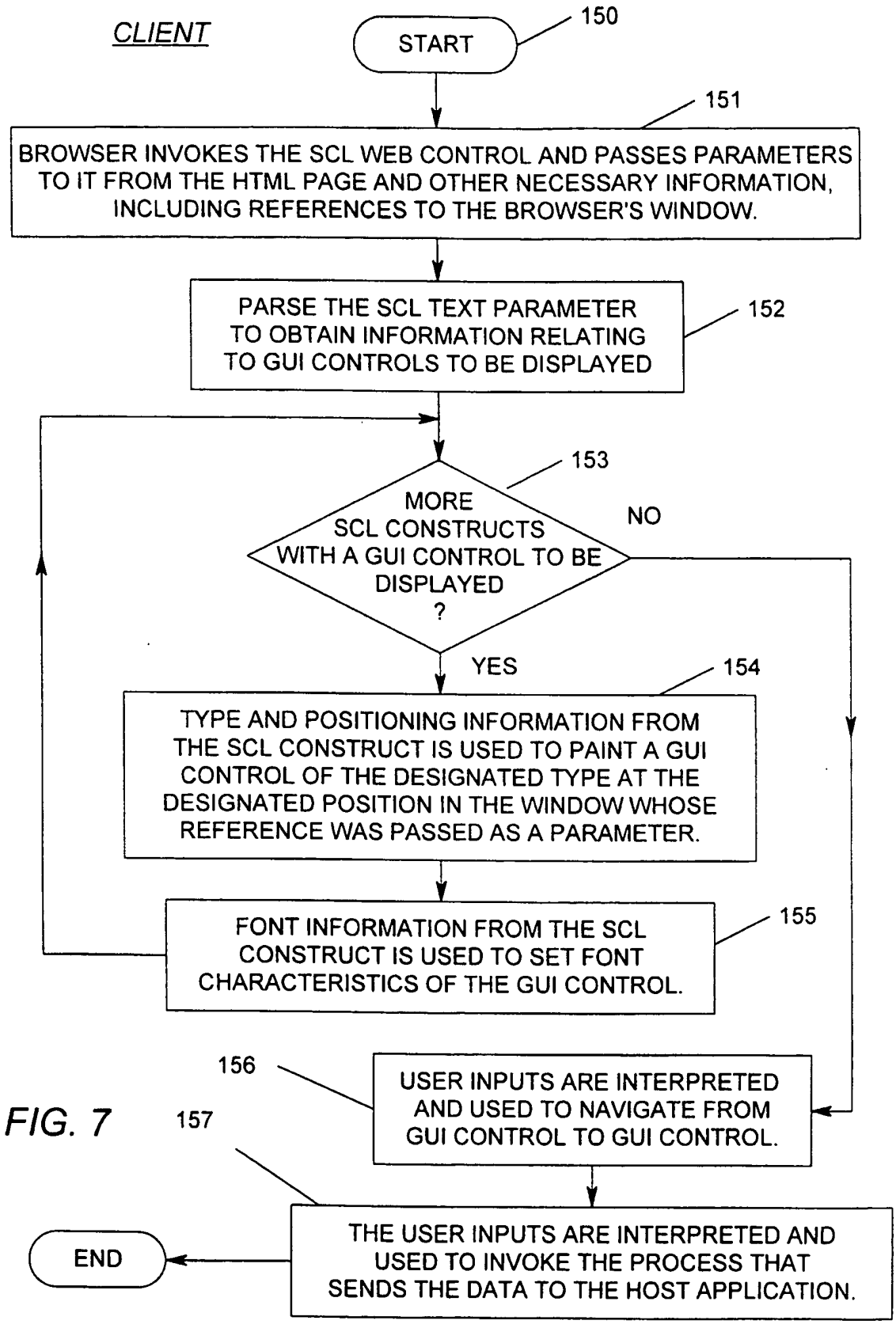

CLIENT
START
150
151
BROWSER INVOKES THE SCL WEB CONTROL AND PASSES PARAMETERS TO IT FROM THE HTML PAGE AND OTHER NECESSARY INFORMATION, INCLUDING REFERENCES TO THE BROWSER'S WINDOW.
PARSE THE SCL TEXT PARAMETER TO OBTAIN INFORMATION RELATING TO GUI CONTROLS TO BE DISPLAYED
152
153
MORE SCL CONSTRUCTS WITH A GUI CONTROL TO BE DISPLAYED ?
NO
YES
154
TYPE AND POSITIONING INFORMATION FROM THE SCL CONSTRUCT IS USED TO PAINT A GUI CONTROL OF THE DESIGNATED TYPE AT THE DESIGNATED POSITION IN THE WINDOW WHOSE REFERENCE WAS PASSED AS A PARAMETER.
FONT INFORMATION FROM THE SCL CONSTRUCT IS USED TO SET FONT CHARACTERISTICS OF THE GUI CONTROL.
155
156
USER INPUTS ARE INTERPRETED AND USED TO NAVIGATE FROM GUI CONTROL TO GUI CONTROL.
157
THE USER INPUTS ARE INTERPRETED AND USED TO INVOKE THE PROCESS THAT SENDS THE DATA TO THE HOST APPLICATION.
END

FIG. 7

INTERNATIONAL SEARCH REPORT

Ir ational Application No
PCT/US 98/20419

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 G06F9/44 G06F17/30

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
P,A	WO 98 06033 A (AGRANAT SYSTEMS INC) 12 February 1998 see abstract see page 5, line 18 - page 6, line 9	1-10
P,A	WO 97 37303 A (OPENCONNECT SYSTEMS INC) 9 October 1997 see abstract; figure 1 see page 2, line 20 - page 5, line 4	1-10
A	WO 96 29663 A (MICROSOFT CORP) 26 September 1996 see abstract; figure 3 see page 10, line 16 - page 11, line 32 see page 16, line 7 - page 19, line 8	1-10

-/--

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
10 February 1999	17/02/1999

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Bowler, A

2

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

In tional Application No
PCT/US 98/20419

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
A	CRESPO A ET AL: "WebWriter: A browser-based editor for constructing Web applications" COMPUTER NETWORKS AND ISDN SYSTEMS, vol. 28, no. 11, May 1996, page 1291-1306 XP004018228 see page 1292 ---	1-10
A	RICHARDS T N: "IBM VISUALAGE CLIENT/SERVER VISUAL PROGRAMMING PRODUCT" PROCEEDINGS SHARE EUROPE MEETING, 26 October 1993, pages 689-707, XP000645957 see page 697 - page 701 ---	1-10
E	US 5 835 712 A (DUFRESNE FRED B) 10 November 1998 see abstract; figure 6 see column 4, line 16 - column 5, line 12 ---	1-10
E	WO 98 47068 A (APPLE COMPUTER) 22 October 1998 see page 13, line 14 - page 19, line 16 -----	1-10

2

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No
PCT/US 98/20419

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 9806033 A	12-02-1998	EP 0853788 A	22-07-1998
WO 9737303 A	09-10-1997	US 5754830 A	19-05-1998
		AU 2597597 A	22-10-1997
WO 9629663 A	26-09-1996	US 5732219 A	24-03-1998
		US 5793966 A	11-08-1998
		EP 0815518 A	07-01-1998
		EP 0815519 A	07-01-1998
		WO 9629664 A	26-09-1996
US 5835712 A	10-11-1998	AU 2826797 A	26-11-1997
		WO 9742584 A	13-11-1997
WO 9847068 A	22-10-1998	NONE	